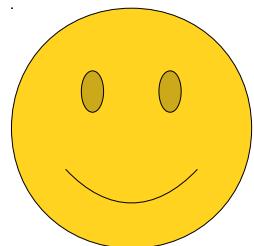
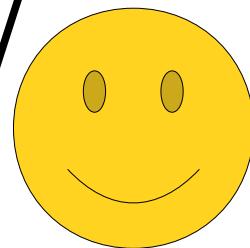


# Assignment 0: Using the Debugger

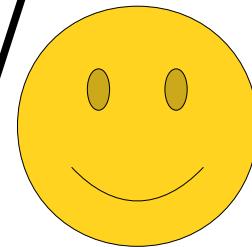


Hi everybody!

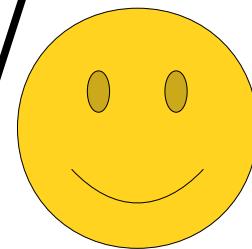
As part of Assignment 0, we'd like you to get a little bit of practice using the debugger in Qt Creator.



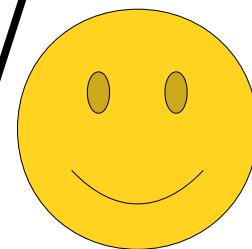
The debugger is a tool you can use to help see what your program is doing as you run it.



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.

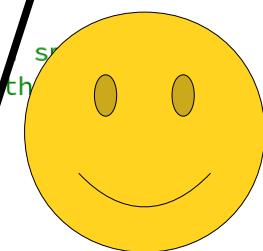


Think of this guide as a little tutorial walkthrough to help give you a sense of how to use the debugger and how to make sense of what you're seeing.



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.

```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
42 * For t
43 * treat
44 * It th
45 * F_p, w
46 * some smaller prime number q. (you aren't expected to
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
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 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 }
```



Activities    Qt Creator ▾

Jan 4 3:09 PM  
NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

42 \* For those of you who are more mathematically inclined, this function  
43 \* treats each character in the input name as a number between 0 and 128.  
44 \* It then uses them as coefficients in a polynomial over the finite field  
45 \*  $F_n$ , where  $n$  is a large prime number, and evaluates that polynomial at  
     $x = 1$ . This is for CS106B,  
    and a small  
    less than

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

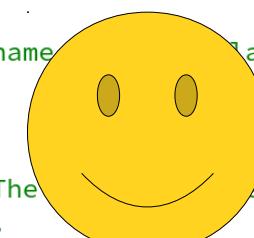
Now, click the mouse!

56 int hashVal = 0;  
57  
58 /\* Iterate across all the characters in the string name, from first to last.  
59 \* name, updating the hash at each step.  
60 \*/  
61 for (char ch: first + last) {  
62 /\* Convert the input character to lower case. The ASCII codes of  
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 }

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities Qt Creator ▾ Jan 4 3:11 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

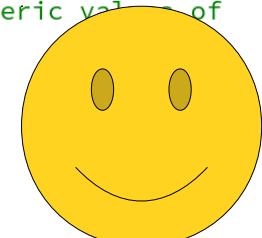
42 \* For those of you who are more mathematically inclined, this function  
43 \* treats each character in the input name as a number between 0 and 128.  
44 \* It then uses them as coefficients in a polynomial over the finite field  
45

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

This is called a **breakpoint**. If we run the program in debug mode, whenever the program gets to this line, it will pause and open up the debugger so we can see what's going on.

60  
61 \*/  
62 for (char ch: first + last) {  
63 /\* Convert the input character to lower case. Lower-case letters are always less than 127.  
64 \*/  
65 ch = tolower(ch);  
66 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
67 }  
68  
69 return hashVal;  
70 }  
71 }

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities    Qt Creator ▾

Jan 4 3:11 PM  
NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

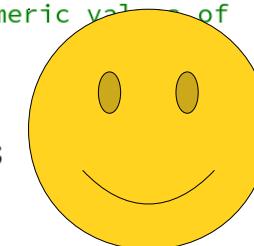
42 \* For those of you who are more mathematically inclined, this function  
43 \* treats each character in the input name as a number between 0 and 128.  
44 \* It then uses them as coefficients in a polynomial over the finite field  
45 \*  $F_p$ , where  $p$  is a large prime number, and evaluates 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  
than

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.

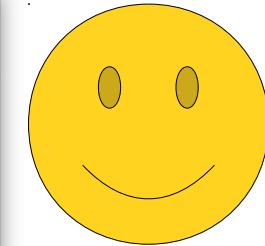
It's the one just below the regular green "run" button. When you do...

61 \*/  
62 for (char ch: first + last) {  
63 /\* Convert the input character to lower case. Lower-case letters have 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 }



Type to locate (Ctrl...)  
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

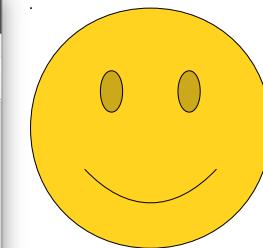
... you should see something like this! Notice that a bunch of extra panels popped up in Qt Creator. We'll talk about what each of these windows mean in a second.



```
File Edit View Build Debug Analyze Tools Window Help  
Activities NameHash  
File Edit View Build Debug Analyze Tools Window Help  
Projects NameHash [main]  
Welcome NameHash.pro  
Sources NameHash.cpp  
Edit  
Design  
Debug  
Projects Help  
NameHash  
Debug  
Run  
Break  
Stop  
Type  
NameHash Console  
File Edit Options Help  
Type  
What is your first name?  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66 cn = tolower(cn);  
67 hashVal = (kSmallPrime * hashVal + ch) % kL  
68 }  
69 return hashVal;  
Debugger GDB for "NameHash" Application started.  
Level Function File Line Address Number Function File Line Address Condition Ignore Threads  
1 ...g) ...eHash.cpp 66 ...5555b6782 (all)  
Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views
```

In the meantime, type in the first name **Ada** and hit enter, as shown here. We specifically want you to enter **Ada** here, *not your actual first name.*  
(Unless your first name is Ada. )

```
43 * some smaller prime number q. (You aren't
44 * but we thought it might be fun!)
45 */
46
47
48
49
50
51
52 What is your first name? Ada
53 What is your last name? |
54
55
56
57
58
59
60
61
62
63
64
65
66     cn = tolower(cn);
67     hashVal = (kSmallPrime * hashVal + ch) % k;
68 }
69 return hashVal;
```



Debugger GDB for "NameHash"										Threads: #12		Application started.		Views	
Level	Function	File	Line	Address	Number	Funct	File	Line	Address	Condition	Ignore	Threads			
					● 1	...g)	...eHash.cpp	66	...5555b6782	(all)					

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

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

Activities NameHash ▾

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] NameHash.pro Sources NameHash.cpp

Welcome Edit Design Debug Projects Help

NameHash

Type

43 \* some smaller prime number q. (You aren't  
44 \* but we thought it might be fun!)  
45 \*/  
46  
47  
48  
49  
50 NameHash Console  
51 File Edit Options Help  
52 What is your first name? Ada  
53 What is your last name? Lovelace  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66 cn = tolower(cn);  
67 hashVal = (kSmallPrime \* hashVal + ch) % k;  
68 }  
69 return hashVal;

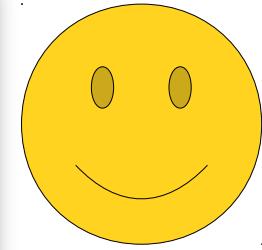
cn = tolower(cn);  
hashVal = (kSmallPrime \* hashVal + ch) % k;  
}

return hashVal;

Debugger GDB for "NameHash" Application started.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
					1	...g)	...eHash.cpp	66	...5555b6782	(all)		

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views



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

Activities NameHash ▾

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] NameHash.pro Sources NameHash.cpp

Welcome Edit Design Debug Projects Help

NameHash

Type

43  
44 \* some smaller prime number q. (You aren't  
45 \* but we thought it might be fun!)  
46 \*/  
47  
48  
49 NameHash Console

File Edit Options Help

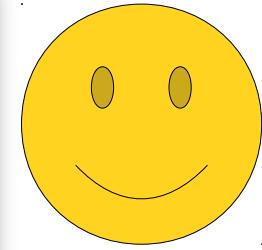
50  
51 What is your first name? Ada  
52 What is your last name? Lovelace  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66 cn = tolower(cn);  
67 hashVal = (kSmallPrime \* hashVal + ch) % k;  
68 }  
69 return hashVal;

cn = tolower(cn);  
hashVal = (kSmallPrime \* hashVal + ch) % k;  
}

Debugger GDB for "NameHash" Application started.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
					1	...g)	...eHash.cpp	66	...5555b6782	(all)		

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views



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

Activities NameHash ▾

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] NameHash.pro Sources NameHash.cpp

Welcome Edit Design Debug Projects Help

NameHash

Debugger GDB for "NameHash" Application started.

Type

NameHash Console

File Edit Options Help

What is your first name? Ada

What is your last name? Lovelace

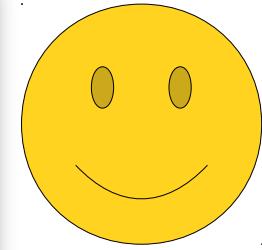
cn = tolower(cn);  
hashVal = (kSmallPrime \* hashVal + ch) % k;  
return hashVal;

Type

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

1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views



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

Qt Creator interface showing a debugger session for a NameHash project.

**Code View:**

```
48
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers.
51      * These numbers were chosen because they are
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case,
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71
```

**Registers View:**

Name	Type	Value
"AdaLovelace"	std::string::iterator	0x555555b6780
'A'	char	65
"Ada"	std::string	0x555555b6784
0	int	0
16908799	int	16908799
127	int	127
"Lovelace"	std::string	0x555555b6788

**Breakpoint 1:** Located at line 66 in NameHash.cpp, where `ch = tolower(ch);` is executed.

**Threads View:**

Number	Function	File	Line	Address	Condition	Ignore	Threads
1	...g) ...eHash.cpp	66	0x555555b6782	0x555555b6782	(all)		

**Bottom Status Bar:**

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

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

Qt Creator - NameHash [main]

File Edit View Build Debug Analyze Tools Window Help

Projects Sources

```
48
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers.
51      * These numbers were chosen because they are
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case.
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```

Breakpoint at line 66

Registers

Name	Type	Value
first	std::string	"AdaLovelace"
last	std::string	"'A'"
hashVal	int	65
kSmallPrime	int	16908799
kLargePrime	int	127
ch	char	'A'

Call Stack

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x555555b6782
2	studentMain	NameHash.cpp	31	0x555555b6595
3	std::function<int ()>::operator() (const QThread*)	QtGui.dll	161	0x5555556161bc
4	GThreadStd::run()	glib-2.0.dll	9476	0x555555f9476
5	??	kernel32.dll	143d84	0x7ffff6143d84
6	start thread	pthread-2.dll	463	0x7ffff257590

Threads: #12 NameHash

Stopped at breakpoint 1 in thread 12.

Views

Type to locate (Ctrl+Shift+F)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Build

Activities Qt Creator ▾ Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .

Name Value Type  
for\_begin @0x7fffc6058c78 std::string::iterator  
for\_end @0x7fffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41

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

62 for (char ch: first + last)  
63 /\* Convert the input characters to  
64 \* lower-case letters are always  
65 \*/  
66 ch = tolower(ch);  
67 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
68 }  
69 return hashVal;  
70 }  
71 }

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12. Level Function File Line Address Number Function File Line Address Condition Ignore Threads  
1 nameHash NameHash.cpp 66 0x555555b6782 1 ...g) ...eHash.cpp 66 ...5555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent  
4 GThreadStd::run()  
5 ??  
6 start\_thread pthread\_create.c 463 0x7ffff6143d84

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views Build

Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41

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.

62 for (char ch: first + last,  
63 /\* Convert the input characters to lower-case letters are always  
64 \* lower-case letters are always  
65 \*/  
66 ch = tolower(ch);  
67 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
68 }  
69 return hashVal;  
70 }  
71 }

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level Function File Line Address  
1 nameHash NameHash.cpp 66 0x555555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent  
4 GThreadStd::run()  
5 ??  
6 start thread pthread\_create.c 463 0x7ffff6143d84

Number Function File Line Address Condition Ignore Threads  
1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views Build

Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      * 2^31 - kLargePrime - 1.
```

Name Value Type  
\_\_for\_begin @0x7ffc6058c78 std::string::iterator  
\_\_for\_end @0x7ffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string &  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41

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

62
63
64
65
66
67
68
69
70
71

```
62     for (char ch: first + last,
63         /* Convert the input characters to
64          * lower-case letters are always
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
```

66

Value Type

Stopped at breakpoint 1 in thread 12.

Level Function File Line Address

1 nameHash NameHash.cpp 66 0x555555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent  
4 GThreadStd::run() 0x555555f9476  
5 ?? 0xfffff6143d84  
6 start thread pthread\_create.c 463 0x7ffff257590

Number Function File Line Address Condition Ignore Threads

1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views Build

Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

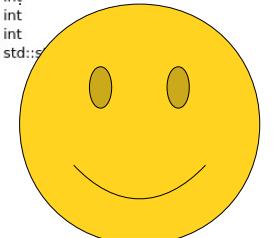
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
\_\_for\_begin @0x7fffc6058c78 std::string::iterator  
\_\_for\_end @0x7fffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string &&  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41



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

Debugger GDB for "NameHash" Threads: #12 NameHash  
Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555b6782	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::function<int ()>::operator() (const QEvent&)	QtGui::startBackgroundEventLoop		0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7fff6143d84								
6	start_thread	pthread.c	463	0x7fff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

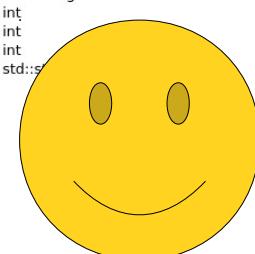
```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      *  $2^{31} - kLargePrime - 1$ .
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime = 127;
56
57     int hashVal = 0;
58
59     /* Iterate across all the characters in the first
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

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

Debugger GDB for "NameHash" Threads: #12 NameHash  
Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555b6782	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7fff6143d84								
6	start_thread	pthread_create.c	463	0x7fff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

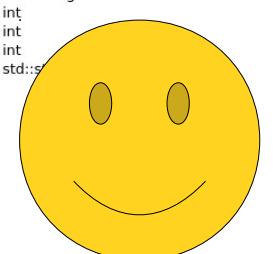
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

nameHash(std::string, std::string) @ Unix (LF) Line: 66, Col: 9

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

0x41



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

Debugger GDB for "NameHash" Threads: #12 NameHash

Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555b6782	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x5555555b6595								
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc								
4	GThreadStd::run()			0x5555555f9476								
5	??			0x7fff6143d84								
6	start_thread	pthread_create.c	463	0x7fff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

The call stack can tell us exactly that!

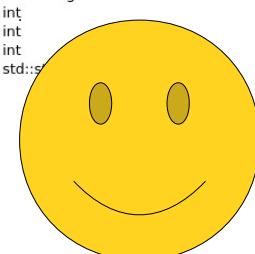
Debugger GDB for "NameHash" Threads: #12 NameHash

Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x555555b6782
2	studentMain	NameHash.cpp	31	0x555555b6595
3	std::function<int ()>::operator() (const std::function<int ()>&, QtGui::QEvent)	0x5555556161bc		
4	GThreadStd::run()	0x555555f9476		
5	??	0xfffff6143d84		
6	start_thread	pthread_create.c	463	0xfffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Build



Activities Qt Creator ▾

Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

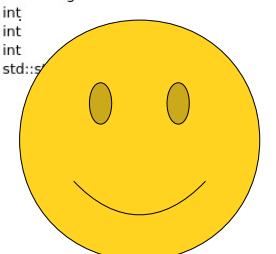
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7fffc6058c78 std::string::iterator  
for\_end @0x7fffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41



Notice that the call stack lists a series of different functions in order. Here, it has `nameHash` (where we are now) at the top, and right below that is `studentMain`.

Debugger GDB for "NameHash" Threads: #12 NameHash  
Level Function File Line Address  
1 nameHash NameHash.cpp 66 0x555555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent...  
4 GThreadStd::run()  
5 ??  
6 start\_thread pthread\_create.c 463 0x7ffff6143d84  
Number Funct File Line Address Condition Ignore Threads  
1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Build

Activities Qt Creator ▾

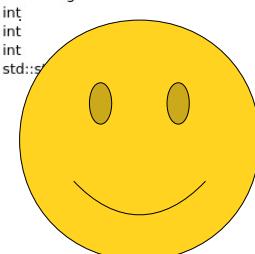
Jan 4 3:15 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch
63         /*
64         *
65         */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7fffc6058c78 std::string::iterator  
for\_end @0x7fffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string && 0x41  
ch 'A' char 65  
first "Ada" std::string int  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string



Go and double-click the call to **studentMain** on Level 2. When you do...

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	main	NameHash.cpp	66	0x555555b6782	1	...g)	...eHash.cpp	66	...5555b6782			(all)
2	studentMain	NameHash.cpp	31	0x5555555b6595								
3	__ZN10QThread11startThread	QThread.h	104	0x5555555f9476								
4	GThreadStart()			0x7ffff6143d84								
5	??											
6	start_thread	pthread.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Build

Activities Qt Creator ▾ Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

NameHash.cpp <Select Symbol> Unix (LF) Line: 31, Col: 5

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * declare it in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash of your name is: " << hashValue;
32     return 0;
33 }
34
35 /* This is the actual function that computes the hash.
   * To talk more about what hash functions do,
   * the meantime, think of it as a function that takes
   * of the input and produces a number.
   *
   * For those of you who are more mathematically inclined,
   * treats each character in the input name as a number
   * and adds them together to produce the hash value.
   */
36
37
38
39
40
41
42
43
```

Name Value Type

first "Ada" std::string  
hashValue 0 int  
last "Lovelace" std::string

You'll end up over here!

Debugger GDB for "NameHash" Threads: #12 NameHash

Level Function File Line Address

1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::Function_handler<int ()>, QWidget::startBackgroundEvent()			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7fff6143d84
6	start_thread	pthread.c	463	0x7fff6257590

Number Function File Line Address Condition Ignore Threads

1	...g) ...eHash.cpp	66	...55555b6782	(all)
---	--------------------	----	---------------	-------

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

NameHash.cpp

19 #include "simpio.h" // for getLine  
20 using namespace std;  
21  
22 /\* Prototype for the nameHash function. This lets us  
 \* put it in main and then define it later in the program.  
 \*/  
23  
24 int nameHash(string first, string last);  
25  
26 int main() {  
27 string first = getLine("What is your first name?  
28 string last = getLine("What is your last name?  
29  
30 int hashValue = nameHash(first, last);  
31  
32 cout << "The hash of your name is: " << hashValue  
33 return 0;  
34 }  
35  
36 /\* This is the actual function that computes  
 \* the hash value. For now, we're just going to  
 \* talk more about what hash functions do  
 \* in the meantime, think of it as a function  
 \* that takes some input and produces a number  
 \* that's unique to that input.  
 \*  
 \* For those of you who are more mathematically  
 \* inclined, this function treats each character in the input  
 \* as a digit in a base-128 number, and then  
 \* converts that to a standard integer.  
 \*/

Line: 31, Col: 5

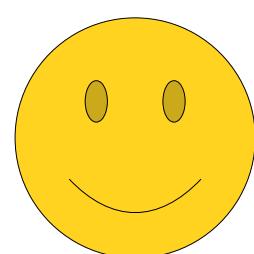
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

Debugger GDB for "NameHash" Threads: #12 NameHash

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x55
2	studentMain	NameHash.cpp	31	0x55
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x55
4	GThreadStd::run()			0x55
5	??			0x7f
6	start thread	pthread_create.c	463	0x7f

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control

Views Threads (all)

 A yellow smiley face icon with a simple black outline, positioned above a speech bubble.

Notice that the yellow arrow points to Line 31. That line 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!

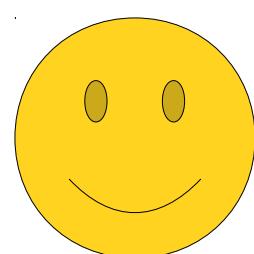
Activities Qt Creator ▾ Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * declare it in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash of your name is: " << hashValue;
32     return 0;
33 }
34
35 /* This is the
   * code to talk more
   * about the meaning
   * of the input
   *
   * For those
   * who treat each character
   * as a separate character
   */
36
37 /* This is the
   * code to talk more
   * about the meaning
   * of the input
   *
   * For those
   * who treat each character
   * as a separate character
   */
38
39
40
41
42
43
```

Name Value Type  
first Ada std::string  
hashValue 0 int  
last Lovelace std::string



Generally speaking, you can use the call stack as a way to see which function calls got us to the point where the program paused at the breakpoint!

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator() (const QWidget*, const QString&)	QtGui::startBackgroundEventLoop		0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views

Activities Qt Creator ▾

Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

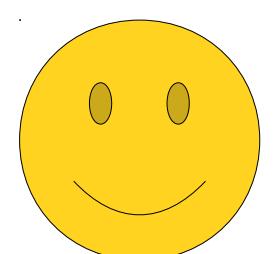
Projects NameHash [main] Sources NameHash.cpp

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash value is " << hashValue << endl;
32
33     return 0;
34 }
35
36 /* This is
   * to talk
   * the meaning
   * of the input and produces a number.
37 */
38
39 /* For those of you who are more mathematically inclined,
   * treats each character in the input name as a number.
40 */
41
42 /*
43 */

  Line: 31, Col: 5
```

Name Value Type

first "Ada" std::string  
hashValue 0 int  
last "Lovelace" std::string



Depending on your OS, you might see some additional functions beneath studentMain. What are those?

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread.c	463	0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾ Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

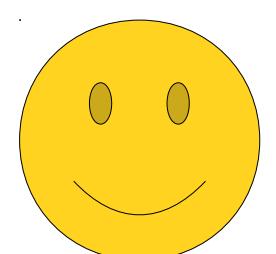
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * declare it here in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << hashValue << endl;
32     return 0;
33 }
34
35 /* This is
   * to talk
   * the means
   * of the input and produces a number.
36
37 *
38 * For those of you who are more mathematically inclined,
39 * treats each character in the input name as a number.
40
41 *
42 * The hash value is the sum of the ASCII values of the characters
43 * in the name, converted to an integer.
44
45 * The ASCII value of a character is its position in the
46 * standard character set, ranging from 0 to 127.
47
48 * For example, the ASCII value of 'A' is 65, so if we
49 * hash the name "Ada", we would get the value 65 + 97 + 100 = 262.
50
51 * Note that this is just one way to hash names. There are many
52 * other ways, and some are better than others for different purposes.
53
54 * One important consideration is that the hash value should be
55 * unique for each name, or at least very unlikely to be the same
56 * for two different names. This is called a "collision-free" hash.
57
58 * Another consideration is that the hash value should be
59 * computationally efficient to calculate, so that it can be done
60 * quickly even for large numbers of names.
61
62 * Finally, it's worth noting that hashing is a fundamental
63 * technique in computer science, used in many different
64 * applications, such as file indexing, password storage, and
65 * data compression.
```

Name Value Type

first "Ada" std::string  
hashValue 0 int  
last "Lovelace" std::string



These grayed-out functions represent helper functions our libraries automatically call to help get your program set up.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread.c	463	0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

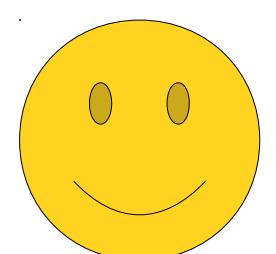
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash value is " << hashValue << endl;
32
33     return 0;
34 }
35
36 /* This is
   * to talk
   * the meant
   * of the input and produces a number.
37 */
38
39 /*
40 * For those of you who are more mathematically inclined,
41 * treats each character in the input name as a number.
42 */
43
```

Name Value Type

first Ada std::string  
hashValue 0 int  
last Lovelace std::string



You don't need to worry about these. They'll show up in all the programs you run and you can safely ignore them.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0xfffff6143d84
6	start_thread	pthread_create.c	463	0xfffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Activities Qt Creator ▾ Jan 4 3:22 PM NameHash.cpp @ NameHash [main] - Qt Creator

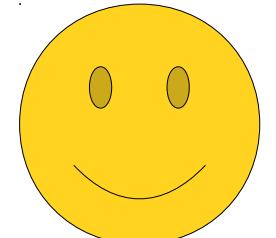
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * declare it in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash value is " << hashValue << endl;
32
33     return 0;
34 }
35
36 /* This is
   * to talk
   * the meant
   * of the input and produces a number.
37 */
38
39 /* For those of you who are more mathematically inclined,
   * treats each character in the input name as a number.
40 */
41
42 /*
43 */

Line: 31, Col: 5
```

Name Value Type  
first "Ada" std::string  
hashValue 0 int  
last "Lovelace" std::string



In the meantime, let's get back to our `nameHash` function. To do that, double-click on the `nameHash` entry at the top of the call stack. When you do...

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level Function File Line Address Number Function File Line Address Condition Ignore Threads (all)

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555b6782	1	...g)	...eHash.cpp	66	...5555b6782			(all)
2	std::				2	operator<	simpio.h	161	0x5555556161bc			
3	std:: Function Handler<int ()>				3	operator<	simpio.h	161	0x555555f9476			
4	GThreadStarter				4	operator<	simpio.h	161	0x7ffff6143d84			
5	??				5	operator<	simpio.h	161	0x7ffff6143d99			
6	start thread				6	operator<	simpio.h	161	0x7ffff6143d99			
						pthread	pthread.c	463	0x7ffff6257590			

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

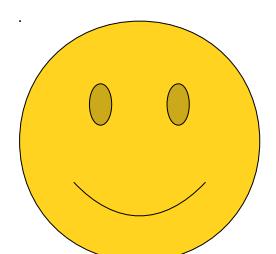
Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch
67         has
68     }
69     return
70 }
71 }
```

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string && 0x41 char  
ch 'A' 65 std::string  
first "Ada" int  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string



You'll be teleported back here!

Debugger GDB for "NameHash" Threads: #12 NameHash  
Level Function File Line Address  
1 nameHash NameHash.cpp 66 0x555555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEve...  
4 GThreadStd::run()  
5 ??  
6 start\_thread pthread\_create.c 463 0x7ffff6143d84  
Number Function File Line Address Condition Ignore Threads  
1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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     /* I
60     * n
61     */
62     for
63
64
65
66     ch = tolower(ch),
67     hashVal = (kSmallPrime * hashVal + ch) % kl
68 }
69
70 return hashVal;
71 }
```

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

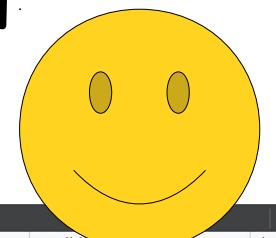
Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
+ 1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x55555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Name	Value	Type
for_begin	@0x7ffc6058c78	std::string::iterator
for_end	@0x7ffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Views (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Line: 66, Col: 9

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
64         * lower-case letters are always less than
65         */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```

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.

66

for (char ch: first + last) {

/\* Convert the input character to lower case  
\* lower-case letters are always less than  
\*/  
ch = tolower(ch);  
hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;

return hashVal;

① 1 ...g) ...eHash.cpp 66 ...5555b6782

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string && 0x41 char  
ch 'A' 65 std::string  
first "Ada" int  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Views

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
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.  
64 \* lower-case letters are always less than  
65 \*/  
66 ch = tolower(ch);  
67 hashVal = (kSmallPrime \* hashVal + ch) % kL  
68 }  
69 return hashVal;  
70 }  
71 }

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

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string && 0x41 char  
ch 'A' 65 std::string  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

Name Value  
Name Hash

Debugger GDB for "NameHash" Threads: #12 NameHash  
Level Function File Line Address  
1 nameHash NameHash.cpp 66 0x555555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent... 0x5555556161bc  
4 GThreadStd::run() 0x555555f9476  
5 ?? 0xfffff6143d84  
6 start\_thread pthread\_create.c 463 0x7ffff6257590  
Number Funct File Line Address Condition Ignore Threads  
1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Line: 66, Col: 9

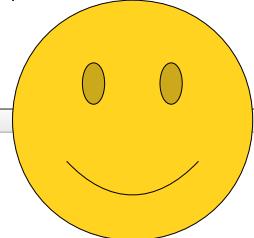
```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      * 2^31 - kLargePrime - 1.
53     */
54
55
56
57     for (char ch: first + last) {
58         /* Convert the input character to lower case
59         * lower-case letters are always less than
60         */
61
62         ch = tolower(ch);
63         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
64     }
65
66     return hashVal;
67 }
68 }
```

The yellow arrow points out where we are right now.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
+ 1	nameHash	NameHash.cpp	66	0x555555b6782
2	studentMain	NameHash.cpp	31	0x555555b6595
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc
4	GThreadStd::run()			0x555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views (all)



Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Line: 66, Col: 9

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      * 2^31 - kLargePrime - 1.
53     */
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71 }
```

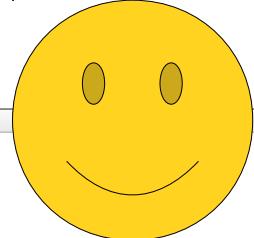
The call stack shows us how we got into the current function.

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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
+ 1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()> QThread::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



0x41

Name	Value	Type
__for_begin	@0x7ffc6058c78	std::string::iterator
__for_end	@0x7ffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Line: 66, Col: 9

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41

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

Smiley face icon

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
+ 1	nameHash	NameHash.cpp	66	0x555555b6782	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start_thread	pthread.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Look up at this panel over here.

Activities    Qt Creator ▾

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

nameHash(string first, string last){  
 /\* This hashing scheme needs two prime numbers.  
 \* prime. These numbers were chosen because the  
 \*  $2^{31} - kLargePrime - 1$ .  
 \*/  
 static const int kLargePrime = 16908799;  
 static const int kSmallPrime = 127;  
  
 int hashVal = 0;  
  
 /\* Iterate across all the characters in the first  
 \* name, updating the hash at each step.  
 \*/  
 for (char ch: first + last) {  
 /\* Convert the input character to lower case.  
 \* lower-case letters are always less than  
 \*/  
 ch = tolower(ch);  
 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
 }  
 return hashVal;  
}

Line: 66, Col: 9

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

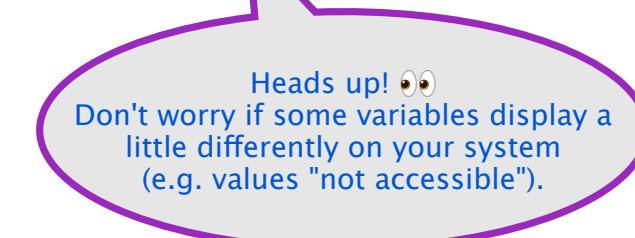
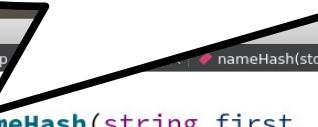
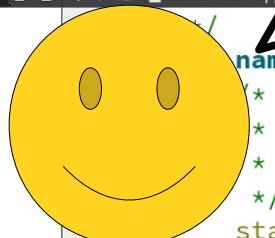
0x41

Name Value Type  
NameHash 66 ...eHash.cpp 66 ...5555b6782

Threads: #12 NameHash  
Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

This window lets you take a look at all the values of the local variables that are in scope right now.



File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

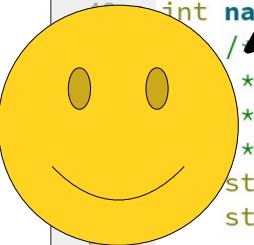
```
nameHash(string first, string last){  
    /* This hashing scheme needs two prime numbers.  
     * prime. These numbers were chosen because they  
     * 2^31 - kLargePrime - 1.  
     */  
    static const int kLargePrime = 16908799;  
    static const int kSmallPrime = 127;  
  
    int hashVal = 0;  
  
    /* Iterate across all the characters in the first  
     * name, updating the hash at each step.  
     */  
    for (char ch: first + last) {  
        /* Convert the input character to lower case.  
         * lower-case letters are always less than  
         */  
        ch = tolower(ch);  
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
    }  
    return hashVal;  
}
```

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
+ 1	nameHash	NameHash.cpp	66	0x555555b6782	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start_thread	pthread_create.c	463	0x7ffff257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Depending on what OS you're using, these might be in a different order, and there might be some weird-looking ones in there in addition to nicer ones like ch and hashVal.



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
NameHash.cpp
48     */
49     int nameHash(string first, string last){
50         /* This hashing scheme needs two prime numbers.
51          * prime. These numbers were chosen because they
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
60          * name, updating the hash at each step.
61         */
62         for (char ch: first + last) {
63             /* Convert the input character to lower case.
64              * lower-case letters are always less than
65              */
66             ch = tolower(ch);
67             hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68         }
69         return hashVal;
70     }
71 }
```

0x41

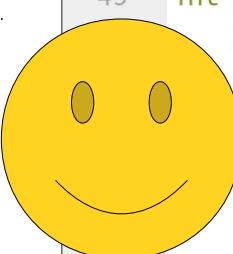
Name	Value	Type
for_begin	@0x7ffc6058c78	std::string::iterator
for_end	@0x7ffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &&
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
+ 1	nameHash	NameHash.cpp	66	0x5555555b6782	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x5555555b6595								
3	std::function<int ()> QtGui::startBackgroundEve...			0x5555556161bc								
4	GThreadStd::run()			0x5555555f9476								
5	??			0x7ffff6143d84								
6	start_thread	pthread.c	463	0x7ffff257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

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



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
48 */ int nameHash(string first, string last){
49 /* This hashing scheme needs two prime numbers.
50 * prime. These numbers were chosen because they
51 * 2^31 - kLargePrime - 1.
52 */
53 static const int kLargePrime = 16908799;
54 static const int kSmallPrime = 127;
55
56 int hashVal = 0;
57
58 /* Iterate across all the characters in the first
59 * name, updating the hash at each step.
60 */
61 for (char ch: first + last) {
62     /* Convert the input character to lower case.
63     * lower-case letters are always less than
64     */
65     ch = tolower(ch);
66     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
67 }
68
69 return hashVal;
70 }
71 }
```

Registers

Name	Type	Value
_for_end	std::string::iterator	0x7ffffc6058c80
_for_range	std::string::iterator	"AdaLovelace"
ch	char	'A'
first	std::string	"Ada"
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

Stack

Name	Type	Value
__func__	std::string	"nameHash"
__PRETTY_FUNCTION__	std::string	"int nameHash(string, string)"
__FUNCTION__	std::string	"nameHash"
__FILE__	std::string	"NameHash.cpp"
__LINE__	int	66

Threads

Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555b6782			(all)
2	studentMain	NameHash.cpp	31	0x5555555b6595			
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc			
4	GThreadStd::run()			0x5555555f9476			
5	??			0x7ffff6143d84			
6	start_thread	pthread_create.c	463	0x7ffff257590			

Stopped at breakpoint 1 in thread 12.

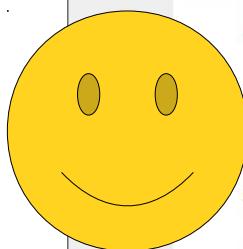
Views

Debugger GDB for "NameHash" Threads: #12 NameHash

Type to locate (Ctrl+Shift+F)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

For example, here you can see the values of `kLargePrime` and `kSmallPrime`, which match the values they were declared with.



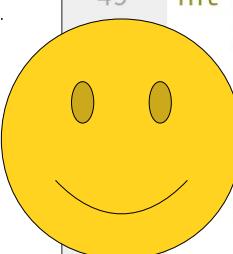


```
48 */  
49 int nam  
50 {  
51     string first, string last){  
52     /* Hashing scheme needs two prime numbers  
53     * prime. These numbers were chosen because they  
54     * 2^31 - kLargePrime - 1.  
55     */  
56     static const int kLargePrime = 16908799,  
57     static const int kSmallPrime = 127;  
58  
59     int hashVal = 0;  
60  
61     /* Iterate across all the characters in the first  
62     * name, updating the hash at each step.  
63     */  
64     for (char ch: first + last) {  
65         /* Convert the input character to lower case.  
66         * Lower-case letters are always less than  
67         */  
68         ch = tolower(ch);  
69         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
70     }  
71     return hashVal;  
72 }
```

```
> __for_end @0x7fffcc6058c80
> __for_range "AdaLovelace"
ch          'A'      65
> first       "Ada"
hashVal     0
kLargePrime 16908799
kSmallPrime 127
> last        "Lovelace"
```

```
type  
std::string::iterator  
std::string::iterator  
std::string &&  
0x41 char  
std::string  
int  
int  
int  
std::string
```

We can also see that, at this point, hashVal is still zero.



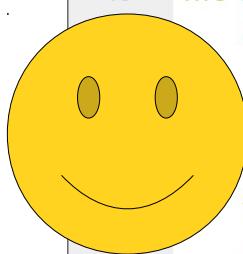
```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
48 */ int nameHash(string first, string last){
49 /* This hashing scheme needs two prime numbers,
50 * prime. These numbers were chosen because they
51 * 2^31 - kLargePrime - 1.
52 */
53 static const int kLargePrime = 16908799;
54 static const int kSmallPrime = 127;
55
56 int hashVal = 0;
57
58 /* Iterate across all the characters in the first
59 * name, updating the hash at each step.
60 */
61 for (char ch: first + last) {
62     /* Convert the input character to lower case
63     * lower-case letters are always less than
64     */
65     ch = tolower(ch);
66     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
67 }
68
69 return hashVal;
70 }
71 }
```

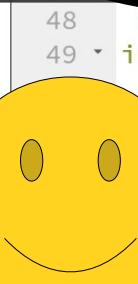
The screenshot shows the Qt Creator IDE interface. The code editor displays the `NameHash.cpp` file with a breakpoint set at line 66. A yellow smiley face icon is overlaid on the code editor area. A blue arrow points from the handwritten note "hashVal is still zero." to the variable declaration `int hashVal = 0;`. The debugger sidebar on the right shows the current state of variables:

Name	Type	Value
__for_end	std::string::iterator	0x7ffffc6058c80
__for_range	std::string::iterator	"AdaLovelace"
ch	char	'A'
first	std::string	"Ada"
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

The status bar at the bottom indicates "Stopped at breakpoint 1 in thread 12." The bottom navigation bar includes tabs for Debugger, GDB for "NameHash", Issues, Search Results, Application Output, Compile Output, QML Debugger Console, Version Control, and Test Results.

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





```
48 */  
49 int nam  
50 {  
51     /* The hashing scheme needs two prime numbers,  
52     * prime. These numbers were chosen because they  
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  
61     * name, updating the hash at each step.  
62     */  
63     for (char ch: first + last) {  
64         /* Convert the input character to lower case.  
65         * lower-case letters are always less than  
66         */  
67         ch = tolower(ch);  
68         hashVal = (kSmallPrime * hashVal + ch) % kL  
69     }  
70     return hashVal;  
71 }
```

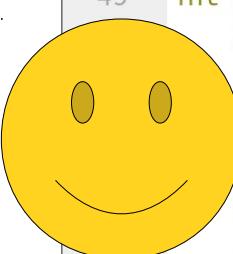
```
> __for_end @0x7fffcb0568c80
> __for_range "AdaLovelace"
ch      'A'   65
> first   "Ada"
hashVal  0
kLargePrime 16908799
kSmallPrime 127
> last    "Lovelace"
```

Name	Value	Type
------	-------	------

Level	Function	File	Line	Address	Number
1	nameHash	NameHash.cpp	66	0x5555555b6782	
2	studentMain	NameHash.cpp	31	0x5555555b6d95	
3	std::function<int ()>::operator()	QtGui::startBackgroundEve...		0x5555556161bc	
4	GThreadStd::run()			0x5555555f9476	
5	??			0x7fff6143d84	
6	start_thread	pthread_create.c	463	0x7fff6257590	

at breakpoint 1 in thread 12.					Views	
Funct	File	Line	Address	Condition	Ignore	Threads
...g)	...eHash.cpp	66	...5555b6782		(all)	

Now, let's take a look at this for loop.



```
48 */  
49 int nameHash(string first, string last){  
    /* This hashing scheme needs two prime numbers,  
     * prime. These numbers were chosen because they  
     * 2^31 - kLargePrime - 1.  
    */  
    static const int kLargePrime = 16908799;  
    static const int kSmallPrime = 127;  
  
    int hashVal = 0;  
  
    /* Iterate across all the characters in the first  
     * name, updating the hash at each step.  
    */  
    for (char ch: first + last) {  
        /* Convert the input character to lower case.  
         * lower-case letters are always less than  
         */  
        ch = tolower(ch);  
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
    }  
    return hashVal;  
}
```

The code is a C++ function named `nameHash` that takes two strings, `first` and `last`, as parameters. It uses two prime numbers, `kLargePrime` and `kSmallPrime`, to calculate a hash value. The function iterates over all characters in both names, converting them to lowercase using `tolower` before updating the hash value. A blue dashed rectangle highlights the loop body from line 66 to line 69.

Registers and Stack:

Name	Type	Value
__for_end	std::string::iterator	0x7fff6058c80
__for_range	std::string::iterator	"AdaLovelace"
ch	char	'A'
first	std::string	"Ada"
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

Breakpoint:

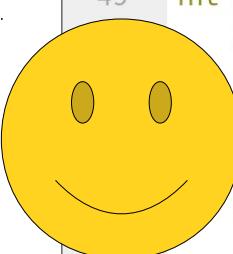
Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
+ 1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator()	QtGui::startBackgroundEve...		0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0xffff6143d84
6	start_thread	pthread_create.c	463	0x7fff6257590

Views:

- Debugger
- GDB for "NameHash"
- Threads: #12 NameHash
- Stopped at breakpoint 1 in thread 12.
- Number
- Funct
- File
- Line
- Address
- Condition
- Ignore
- Threads

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



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
48  /* Hash function
49  int nameHash(string first, string last){
50      /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because they
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
60      * name, updating the hash at each step.
61      */
62      for (char ch: first + last) {
63          /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66          ch = tolower(ch);
67          hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68      }
69
70      return hashVal;
71 }
```

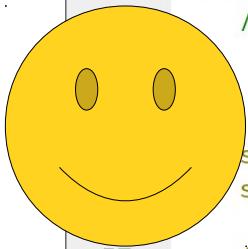
The code is a C++ function named `nameHash` that takes two strings, `first` and `last`, as parameters. It uses two prime numbers, `kLargePrime` and `kSmallPrime`, to calculate a hash value. The function iterates over each character in the concatenated string `first + last`, converts it to lowercase using `tolower`, and updates the hash value using the formula  $\text{hashVal} = (\text{kSmallPrime} * \text{hashVal} + \text{ch}) \% \text{kLargePrime}$ .

The code editor shows a breakpoint at line 66, where `ch` is being converted to lowercase. The debugger window shows the current state of variables:

Name	Type	Value
<code>_for_end</code>	<code>std::string::iterator</code>	<code>@0x7fffcc6058c80</code>
<code>_for_range</code>	<code>std::string</code>	<code>"AdaLovelace"</code>
<code>ch</code>	<code>char</code>	<code>'A'</code>
<code>first</code>	<code>std::string</code>	<code>"Ada"</code>
<code>hashVal</code>	<code>int</code>	<code>0</code>
<code>kLargePrime</code>	<code>int</code>	<code>16908799</code>
<code>kSmallPrime</code>	<code>int</code>	<code>127</code>
<code>last</code>	<code>std::string</code>	<code>"Lovelace"</code>

The debugger also shows the call stack and registers.

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



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
48     */
49     int nameHash(string first, string last){
50         /* This hashing scheme needs two prime numbers,
51          * prime. These numbers were chosen because they
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
60          * name, updating the hash at each step.
61         */
62         for (char ch: first + last) {
63             /* Convert the input character to lower case
64              * lower-case letters are always less than
65              */
66             ch = tolower(ch);
67             hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68         }
69
70     }
71 }
```

The code implements a rolling hash function. It takes two strings, `first` and `last`, and initializes a hash value `hashVal` to 0. It then iterates over each character in the concatenated string `first + last`, converts it to lowercase using `tolower`, and updates the hash value by multiplying it with `kSmallPrime` and adding the character's ASCII value. Finally, it returns the hash value.

Registers:

Name	Type	Value
__for_end	std::string::iterator	0x7ffffc6058c80
__for_range	std::string::iterator	"AdaLovelace"
ch	char	'A'
first	std::string	"Ada"
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

Stack:

Name	Value	Type
hashVal	0x5555555b6782	int

Call Stack:

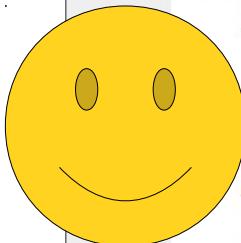
Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int ()>::QtGui::startBackgroundEvent()			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0xfffff6143d84
6	start_thread	pthread_create.c	463	0x7ffffc257590

Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

If we take a look at the current value of the variable `ch`, we can see that it has the value A. That's the first letter of the name Ada Lovelace.



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
48     */
49     int nameHash(string first, string last){
50         /* This hashing scheme needs two prime numbers,
51          * prime. These numbers were chosen because they
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
60          * name, updating the hash at each step.
61         */
62         for (char ch: first + last) {
63             /* Convert the input character to lower case
64              * lower-case letters are always less than
65              */
66             ch = tolower(ch);
67             hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68         }
69         return hashVal;
70     }
71 }
```

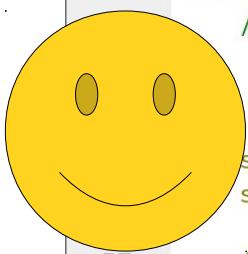
The code is a C++ function named `nameHash` that takes two strings, `first` and `last`, as parameters. It uses two prime numbers, `kLargePrime` and `kSmallPrime`, to calculate a hash value. The function iterates over each character in the concatenated string (`first + last`) and updates the hash value by multiplying it with `kSmallPrime` and adding the ASCII value of the character converted to lowercase (`tolower(ch)`). The final hash value is returned.

Breakpoint 1 is set on line 66, where `ch` is assigned the value 'A'. The debugger shows the current state of variables:

Name	Type	Value
for_end	std::string::iterator	0x7fff6058c80
for_range	std::string::iterator	"AdaLovelace"
ch	char	'A'
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

The debugger also shows the stack trace and threads information.

So now we know where we are (line 66), how we got there (main called nameHash), and the values in the program at this point.



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.cpp
48     */
49     int nameHash(string first, string last){
50         /* This hashing scheme needs two prime numbers,
51          * prime. These numbers were chosen because they
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
60          * name, updating the hash at each step.
61         */
62         for (char ch: first + last) {
63             /* Convert the input character to lower case
64              * lower-case letters are always less than
65              */
66             ch = tolower(ch);
67             hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68         }
69         return hashVal;
70     }
71 }
```

The code is a C++ function named nameHash that takes two strings, first and last, and returns an integer hash value. It uses two prime numbers, kLargePrime (16908799) and kSmallPrime (127). The function iterates over all characters in both strings, converting them to lowercase using tolower, and updates a hash value by multiplying it with kSmallPrime and adding the character's ASCII value. A yellow smiley face icon is overlaid on the code editor area.

Registers

Name	Type	Value	Type
_for_end	std::string::iterator	@0x7fffcc6058c80	
_for_range	std::string::iterator	"AdaLovelace"	
ch	char	'A'	65
first	std::string	"Ada"	
hashVal	int	0	
kLargePrime	int	16908799	
kSmallPrime	int	127	
last	std::string	"Lovelace"	

Stack

Name	Value	Type
hashVal	0	int

Call Stack

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator()()	QtGui::startBackgroundEventLoop		0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0xfffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff257590

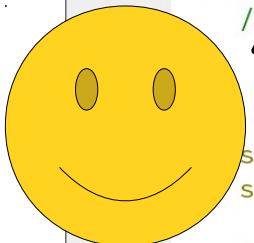
Threads: #12 NameHash

Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Now, let's do something really cool - we're going to run this program one line at a time, watching what happens at each step!



```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
Sources NameHash.pro
NameHash.cpp
48  /* Hash function
49  int nameHash(string first, string last){
50      /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because they
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
60      * name, updating the hash at each step.
61      */
62      for (char ch: first + last) {
63          /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66          ch = tolower(ch);
67          hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68      }
69
70      return hashVal;
71 }
```

The code is a C++ implementation of a rolling hash function. It takes two strings, `first` and `last`, and calculates a hash value. The hash is initialized to 0. It then iterates over each character in the concatenated string `first + last`, converts it to lowercase using `tolower`, and updates the hash value by multiplying it by `kSmallPrime` and adding the character's ASCII value. Finally, it returns the hash value.

Breakpoint 1 is set on line 66, where `ch = tolower(ch);` is executed. The debugger shows the current state of variables:

Name	Type	Value
for_end	std::string::iterator	0x7fff6058c80
for_range	std::string::iterator	"AdaLovelace"
ch	char	'A'
first	std::string	"Ada"
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

The status bar at the bottom indicates "Stopped at breakpoint 1 in thread 12".

Activities Qt Creator ▾

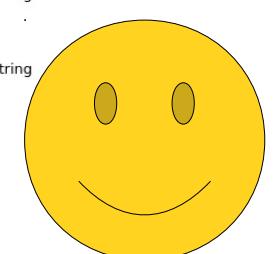
Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
\_\_for\_begin @0x7fffc6058c78 std::string::iterator  
\_\_for\_end @0x7fffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string && 0x41  
ch 'A' char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string



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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

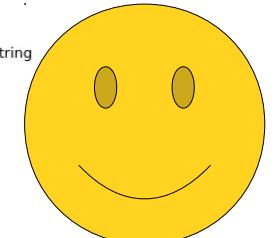
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41



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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

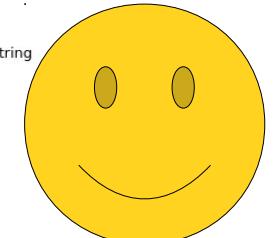
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

0x41

Name	Value	Type
for_begin	@0x7fffc6058c78	std::string::iterator
for_end	@0x7fffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string



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."

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Activities Qt Creator ▾

Jan 4 3:30 PM NameHash.cpp @ NameHash [main] - Qt Creator

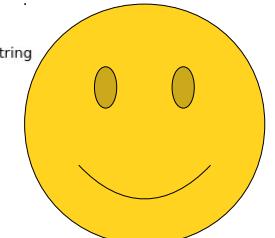
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'A' 65 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x41



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...

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Activities Qt Creator ▾

Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

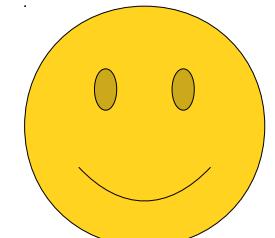
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

Debugger GDB for "NameHash"

Name	Value	Type
for_begin	@0x7fffc6058c78	std::string::iterator
for_end	@0x7fffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

...your window should look something like this.



Activities Qt Creator ▾

Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

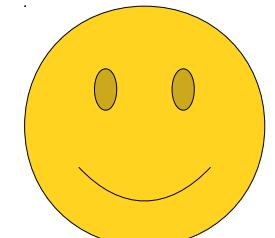
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

Debugger GDB for "NameHash"

Name	Value	Type
for_begin	@0x7ffc6058c78	std::string::iterator
for_end	@0x7ffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

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



Activities Qt Creator ▾ Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

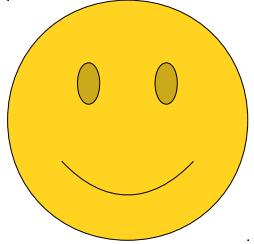
```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      *  $2^{31} - kLargePrime - 1$ .
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime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

Debugger GDB for "NameHash"

Name	Value	Type
__for_begin	@0x7ffc6058c78	std::string::iterator
__for_end	@0x7ffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

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



Activities    Qt Creator ▾

Jan 4 3:42 PM  
NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main]  
Sources NameHash.cpp

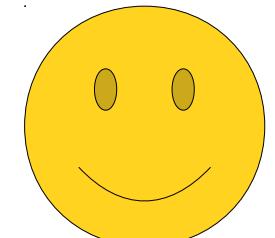
```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      *  $2^{31} - kLargePrime - 1$ .
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime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

Debugger GDB for "NameHash"  
Level Function  
1 nameHash  
2 studentMain  
3 std::Function\_handler<int ()>  
4 GThreadStd::run()  
5 ??  
6 start thread

pthread\_create.c 463 0x7ffff6257590

Name Value Type  
\_\_for\_begin @0x7fffc6058c78 std::string::iterator  
\_\_for\_end @0x7fffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string &&  
ch 'a' 97 char  
first "Ada" std::string  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x61



We're now at the line right after the one where we stopped. You just ran a single line of the program! Pretty cool!

Activities Qt Creator ▾

Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

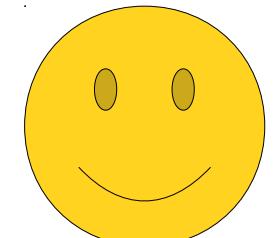
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

Debugger GDB for "NameHash"

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

So what did that line of code do?



Activities Qt Creator ▾ Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

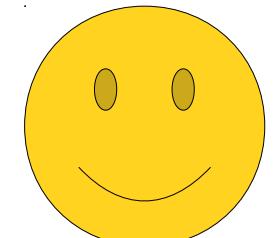
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

Debugger GDB for "NameHash"

Name	Value	Type
for_begin	@0x7fffc6058c78	std::string::iterator
for_end	@0x7fffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

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



You can actually see this by looking at the values panel over on the side!



```
int nameHash(string first, string last){  
    /* This hashing scheme needs two prime numbers,  
     * prime. These numbers were chosen because the  
     * 2^31 - kLargePrime - 1.  
     */  
    static const int kLargePrime = 16908799;  
    static const int kSmallPrime = 127;  
  
    int hashVal = 0;  
  
    /* Iterate across all the characters in the first  
     * name, updating the hash at each step.  
     */  
    for (char ch: first + last) {  
        /* Convert the input character to lower case.  
         * lower-case letters are always less than  
         */  
        ch = tolower(ch);  
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
    }  
    return hashVal;  
}
```

The screenshot shows the Qt Creator IDE interface during a debug session. The code editor displays the `nameHash` function. A callout bubble from the text above points to the "Values" panel on the right, which shows the current state of variables. The variable `hashVal` is highlighted in the code and has its value (0) displayed in the panel. The "Registers" panel below it shows the CPU register state.

Name	Value	Type
for_begin	@0x7ffc6058c78	std::string::iterator
for_end	@0x7ffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'a'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Number	Function	File	Line	Address	Condition	Ignore	Threads
1	...g) ...eHash.cpp	66		...5555b6782	(all)		

At the bottom, the status bar shows the message "Stopped: 'end-stepping-range'".

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



Qt Creator interface showing a debugger session for a C++ project named "NameHash".

The code in the editor is:

```
int nameHash(string first, string last){  
    /* This hashing scheme needs two prime numbers,  
     * prime. These numbers were chosen because the  
     * 2^31 - kLargePrime - 1.  
     */  
    static const int kLargePrime = 16908799;  
    static const int kSmallPrime = 127;  
  
    int hashVal = 0;  
  
    /* Iterate across all the characters in the first  
     * name, updating the hash at each step.  
     */  
    for (char ch: first + last) {  
        /* Convert the input character to lower case  
         * lower-case letters are always less than  
         */  
        ch = tolower(ch);  
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
    }  
    return hashVal;  
}
```

The debugger's registers window shows:

Name	Type	Value
for_begin	std::string::iterator	0x7fffc6058c78
for_end	std::string::iterator	0x7fffc6058c80
for_range	std::string &	Ada Lovelace
ch	char	'a'
first	std::string	"Ada"
hashVal	int	0
kLargePrime	int	16908799
kSmallPrime	int	127
last	std::string	"Lovelace"

The stack window shows:

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x555555b6790
2	studentMain	NameHash.cpp	31	0x555555b6595
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc
4	GThreadStd::run()			0x555555f9476
5	??			0xfffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff257590

The bottom status bar shows: Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

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



```
int nameHash(string first, string last){  
    /* This hashing scheme needs two prime numbers,  
     * prime. These numbers were chosen because the  
     * 2^31 - kLargePrime - 1.  
     */  
    static const int kLargePrime = 16908799;  
    static const int kSmallPrime = 127;  
  
    int hashVal = 0;  
  
    /* Iterate across all the characters in the first  
     * name, updating the hash at each step.  
     */  
    for (char ch: first + last) {  
        /* Convert the input character to lower case.  
         * lower-case letters are always less than  
         */  
        ch = tolower(ch);  
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
    }  
    return hashVal;  
}
```

Qt Creator Debug View

Name	Value	Type
for_begin	@0x7ffc6058c78	std::string::iterator
for_end	@0x7ffc6058c80	std::string::iterator
for_range	Ada lovelace" 97	std::string &
ch	'a'	char
first	"Ada	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range". Views

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	67	0x555555b6790	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::Function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start thread	pthread_create.c	463	0x7ffff257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

This indicates that the value here has changed since the previous step. This is a really useful way to keep track of what's changing as you run the program.



```
  int nameHash(string first, string last){  
 50     /* This hashing scheme needs two prime numbers  
 51         * prime. These numbers were chosen because the  
 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  
 60         * name, updating the hash at each step.  
 61         */  
 62     for (char ch: first + last) {  
 63         /* Convert the input character to lower case.  
 64             * Lower-case letters are always less than  
 65             */  
 66     ch = tolower(ch);  
 67     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
 68 }  
 69  
 70 }
```

	Value	Type
▸ <code>_for_begin</code>	<code>@0x7ffc6058c78</code>	<code>std::string::iterator</code>
▸ <code>_for_end</code>	<code>@0x7ffc6058c80</code>	<code>std::string::iterator</code>
▸ <code>_for_range</code>	<code>Ada Lovelace"</code>	<code>std::string &amp;</code>
<code>ch</code>	<code>'A'</code>	<code>char</code>
<code>first</code>	<code>"Ada"</code>	<code>std::string</code>
<code>hashVal</code>	<code>0</code>	<code>int</code>
<code>kLargePrime</code>	<code>16908799</code>	<code>int</code>
<code>kSmallPrime</code>	<code>127</code>	<code>int</code>
▸ <code>last</code>	<code>"Lovelace"</code>	<code>std::string</code>

Name	Value	Type
------	-------	------

Activities Qt Creator ▾ Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

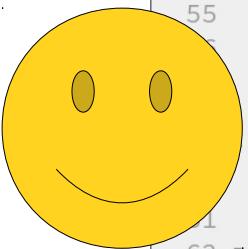
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Welcome Edit Design Debug Projects Help

NameHash

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



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

Start

```
int hashVal = 0;

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

Type

Name	Value	Type
...g)	...eHash.cpp	66
...5555b6782		

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	67	0x555555b6790	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::function<int ()>, QtGui::startBackgroundEve...			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾ Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Not gonna lie, this is a pretty dense line of code. It performs some weird sort of mathematical calculation on a bunch of different values.

Start

```
54
55
56
57
58
59
60
61
62
63
64
65
● 66
⇒ 67 hashVal = 0;
68
69
70
71 }
```

/\* Iterate across all the characters in the first name, updating the hash at each step. \*/  
for (char ch: first + last) {  
 /\* Convert the input character to lower case.  
 \* lower-case letters are always less than  
 \*/  
 ch = tolower(ch);  
 hashVal = (kSmallPrime \* hashVal + ch) % k;

return hashVal;

Name Value Type

Level Function File Line Address

1 nameHash NameHash.cpp 67 0x555555b6790  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent  
4 GThreadStd::run()  
5 ??  
6 start thread pthread\_create.c 463 0x7fffff257590

Number Function File Line Address Condition Ignore Threads

1 ...g) ...eHash.cpp 66 ...5555b6782 (all)

Type  
::string::iterator  
::string::iterator  
::string &&  
ar  
::string  
::string

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results Views

Activities Qt Creator ▾ Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

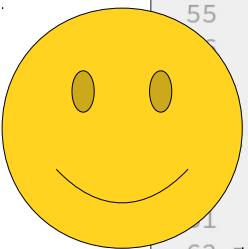
Welcome Edit Design Debug Projects Help

NameHash

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range". Views

Type  
::string::iterator  
::string::iterator  
::string &&  
char  
::string  
::string

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



```
54  
55 Start  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67 hashVal = 0;  
68 /* Iterate across all the characters in the first name, updating the hash at each step.  
69 */  
70 for (char ch: first + last) {  
71     /* Convert the input character to lower case  
72      * lower-case letters are always less than  
73      */  
74     ch = tolower(ch);  
75     hashVal = (kSmallPrime * hashVal + ch) % k;  
76 }  
77 return hashVal;  
78 }
```

Name Value Type

Name	Value	Type
hashVal	0	int

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8

Activities Qt Creator ▾ Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Welcome Edit Design Debug Projects Help

NameHash

54  
55 Start  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67 hashVal = 0;  
68 /\* Iterate across all the characters in the first name, updating the hash at each step.  
69 \*/  
70 for (char ch: first + last) {  
71 /\* Convert the input character to lower case, lower-case letters are always less than  
72 \* uppercase letters.  
73 \*/  
74 ch = tolower(ch);  
75 hashVal = (kSmallPrime \* hashVal + ch) % k;  
76 }  
77 return hashVal;

Let's go run that line of code and see what happens!

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	67	0x555555b6790	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::function<int ()>, QtGui::startBackgroundEve...			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:42 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
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  
60 \* name, updating the hash at each step.  
61  
62  
63  
64  
65  
66 ● 66  
67 ➤ 67  
68  
69 return hashVal;  
70 }  
71

Line: 67, Col: 9

Name Value Type  
for\_begin @0x7fffc6058c78 std::string::iterator  
for\_end @0x7fffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string && 0x61  
ch 'a' char 97  
first "Ada" std::string int  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x61

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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

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

1 nameHash NameHash.cpp 67 0x555555b6790 ● 1 ...g) ...eHash.cpp 66 ...5555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent...  
4 GThreadStd::run()  
5 ??  
6 start\_thread pthread\_create.c 463 0x7ffff6143d84  
0x555555f9476  
0xfffff6143d84  
pthread\_create.c 463 0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views



Activities Qt Creator ▾

Jan 4 3:48 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```

Line: 62, Col: 5

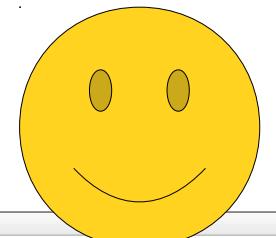
Name	Value	Type
__for_begin	@0x7ffc6058c78	std::string::iterator
__for_end	@0x7ffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &
ch	'a'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Debugger GDB for "NameHash" Level Function

1 nameHash  
2 studentMain  
3 std::Function\_handler<int ()>::operator()<QtGui::startBackgroundEventLoop>() const@0x555555501610L  
4 GThreadStd::run()  
5 ??  
6 start\_thread

pthread\_create.c 463 0x7ffff6143d84

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



Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Activities Qt Creator ▾

Jan 4 3:48 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```

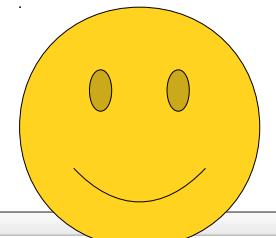
Name Value Type  
\_\_for\_begin @0x7ffc6058c78 std::string::iterator  
\_\_for\_end @0x7ffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string &  
ch 'a' 97 0x61 char  
first "Ada" 97 int  
hashVal 97 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

Debugger GDB for "NameHash" Level Function

1 nameHash  
2 studentMain  
3 std::Function\_handler<int ()>::operator()<QtGui::startBackgroundEventLoop>() const@0x555555501610L  
4 GThreadStd::run()@0x5555555f9476  
5 ??@0xfffff6143d84  
6 start\_thread@0xfffff6257590

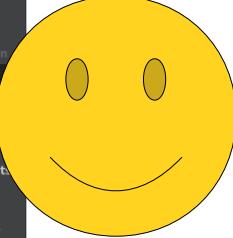
pthread\_create.c 463 0x7ffff6257590

Type to locate (Ctrl+Shift+F) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8



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!



```
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     *  $2^{31} - kLargePrime - 1$ .
     */
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;

int hashVal = 0;

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

The screenshot shows the Qt Creator IDE interface. On the left is the project tree for "NameHash [main]" with files "NameHash.pro" and "NameHash.cpp". The code editor shows the provided C++ code. A callout bubble points from the handwritten note to the variable declaration "int hashVal = 0;" in line 57. To the right of the code editor is the debugger's registers window. The register "hashVal" is highlighted with a blue box and has its value "97" displayed in red, matching the color of the handwritten note. The debugger also shows other variables like "kLargePrime" and "kSmallPrime" with their values and types. At the bottom, the GDB stack trace shows the current thread is stopped at line 66 of "eHash.cpp".

Activities Qt Creator ▾ Jan 4 3:48 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

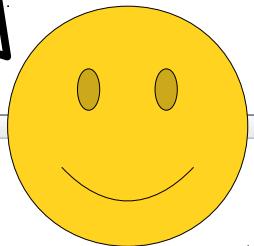
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      *  $2^{31} - kLargePrime = 1$ 
53      */
54     static int has;
55     static /* Iter
56     * name
57     */
58     for (char ch: first + last) {
59         /* Convert the input character to lower case
50      * lower-case letters are always less than
51      */
60         ch = tolower(ch);
61         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
62     }
63     return hashVal;
64 }
```

Line: 62, Col: 5

Name	Value	Type
__for_begin	@0x7ffc6058c78	std::string::iterator
__for_end	@0x7ffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'a'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	16908799	int
kSmallPrime	127	int

Second, notice that we're back up at the top of the for loop, since that's where the yellow arrow is pointing. We ended up back here because this is the next line that gets executed.



Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	62	0x555555b67cb	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start_thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾ Jan 4 3:48 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

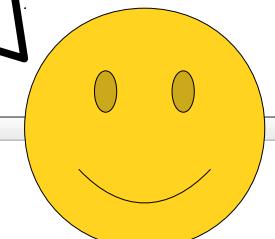
48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
 \* prime. These numbers were chosen because the  
 \*  $2^{31} - kLargePrime - 1$ .  
 \*/  
51 static const int kLargePrime = 16908799;  
52 static const int kSmallPrime = 127;  
53  
54 int hashVal = 1;  
55 /\* Iterates over each character in the string  
 \* name, starting at first and ending at last.  
 \*/  
56 for (char ch: first + last) {  
57 /\* Convert the input character to lower case.  
 \* Lower-case letters are always less than  
 \*/  
58 ch = tolower(ch);  
59 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
60 }  
61 return hashVal;  
62 }

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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	62	0x555555b67cb	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::function<int ()>	QtGui::startBackgroundEve...		0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities Qt Creator ▾ Jan 4 3:48 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

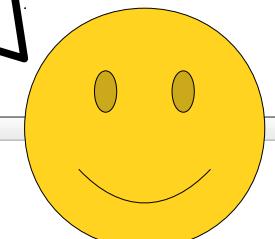
Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      *  $2^{31} - kLargePrime - 1$ .
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime = 127;
56
57     int hashVal = 1;
58
59     /* Iterates over each character in the string
60      * name.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case.
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

Line: 62, Col: 5

Name	Value	Type
for_begin	@0x7fffc6058c78	std::string::iterator
for_end	@0x7fffc6058c80	std::string::iterator
for_range	"AdaLovelace"	std::string &
ch	'a'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Let's go do it again!



Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555b67cb	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x5555555b6595								
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc								
4	GThreadStd::run()			0x5555555f9476								
5	??			0x7ffff6143d84								
6	start thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾

Jan 4 3:48 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      * 2^31 - kLargePrime - 1.
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime = 127;
56
57     int hashVal = 1;
58
59     /* Iterates over every character in the string
60      * name.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case.
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

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

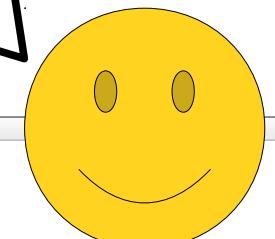
Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Number Function File Line Address

Number	Function	File	Line	Address
1	nameHash	NameHash.cpp	62	0x555555b67cb
2	studentMain	NameHash.cpp	31	0x555555b6595
3	std::function<int ()>::operator()()	...eHash.cpp	66	0x5555556161bc
4	GThreadStd::run()	0x555555f9476	0xfffff6143d84	0x7ffff6143d84
5	??	pthread_create.c	463	0x7ffff6257590

Views



Activities Qt Creator ▾

Jan 4 3:50 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      *  $2^{31} - kLargePrime - 1$ .
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime = 127;
56
57     int hashVal = 97;
58
59     /* Iterators for the strings.
60      * nameHash is the first character of the first string.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case.
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```

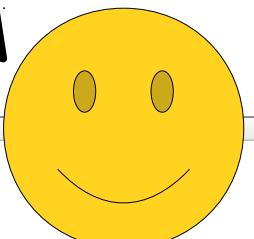
Now we're here! Notice that ch now has the value 'd', which is the second letter of the name Ada.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x555555b6782
2	studentMain	NameHash.cpp	31	0x555555b6595
3	std::function<int ()>::operator()	QtGui::startBackgroundEventLoop		0x5555556161bc
4	GThreadStd::run()			0x555555f9476
5	??			0xfffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Number	Function	File	Line	Address	Condition	Ignore	Threads
1	...g() ...eHash.cpp	66		...5555b6782	(all)		

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Activities    Qt Creator ▾

Jan 4 3:50 PM  
NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main]  
Sources NameHash.pro NameHash.cpp

Line: 66, Col: 9

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      * 2^31 - kLargePrime - 1.
53      */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime = 127;
56
57     int hashVal = 1;
58
59     /* Iterates over every character in the string
60      * name.
61      */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case.
64          * lower-case letters are always less than
65          */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

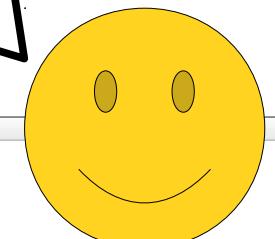
Name Value Type  
\_\_for\_begin @0x7fffc6058c78 std::string::iterator  
\_\_for\_end @0x7fffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string &&  
ch 'd' 100 char  
first "Ada" std::string  
hashVal 97 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

0x64

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

Debugger GDB for "NameHash" Threads: #12 NameHash  
Stopped at breakpoint 1 in thread 12.

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views



Activities Qt Creator ▾

Jan 4 3:52 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

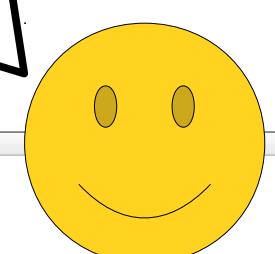
48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .  
53 \*/  
54 static const int kLargePrime = 16908799;  
55 static const int kSmallPrime = 127;  
56  
57 int hashVal = 97;  
58  
59 /\* Iterates over each character in the string.  
60 \* nameHash converts all characters to lower case.  
61 \*/  
62 for (char ch: first + last) {  
63 /\* Convert the input character to lower case.  
64 \* lower-case letters are always less than  
65 \*/  
66 ch = tolower(ch);  
67 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
68 }  
69  
70 return hashVal;  
71 }

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'.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	67	0x555555b6790	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::function<int ()>	QtGui::startBackgroundEventLoop()		0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start_thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities Qt Creator ▾ Jan 4 3:52 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .  
53 \*/  
54 static const int kLargePrime = 16908799;  
55 static const int kSmallPrime = 127;  
56  
57 int hashVal = 97;  
58 /\* Iterates over each character in the string.  
59 \* nameHash converts all characters to lower case.  
60 \* lower-case letters are always less than  
61 \* upper-case letters.  
62 \*/  
63 for (char ch: first + last) {  
64 /\* Convert the input character to lower case.  
65 \* lower-case letters are always less than  
66 \* upper-case letters.  
67 \*/  
68 ch = tolower(ch);  
69 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
70 }  
71  
72 return hashVal;

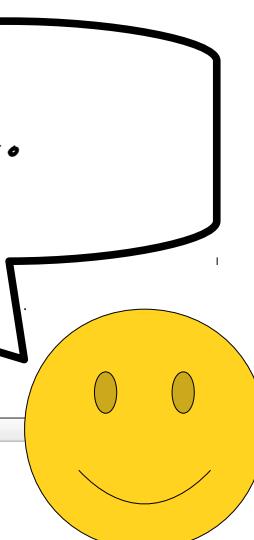
Name Value Type  
\_\_for\_begin @0x7ffc6058c78 std::string::iterator  
\_\_for\_end @0x7ffc6058c80 std::string::iterator  
\_\_for\_range "AdaLovelace" std::string &  
ch 'd' 100 0x64 char  
first "Ada" std::string  
hashVal 97 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

Now, click "Step Over" one more time.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range". Views

Level Function File Line Address Number Function File Line Address Condition Ignore Threads  
1 nameHash NameHash.cpp 67 0x555555b6790 ● 1 ...g) ...eHash.cpp 66 ...5555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBarEndEvent... 0x5555556161bc  
4 GThreadStd::run() 0x555555f9476  
5 ?? 0xfffff6143d84  
6 start thread pthread\_create.c 463 0xfffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Activities Qt Creator ▾

Jan 4 3:53 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .  
53 \*/  
54 static const int kLargePrime = 16908799;  
55  
56  
57  
58 }  
59  
60  
61  
62 }  
63  
64 return hashVal;  
65 }  
66  
67  
68 }  
69  
70 }  
71 }

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &  
ch 'd' char  
first "Ada" 0x64 std::string  
hashVal ??? 0x64  
kLargePrime 16908799  
kSmallPrime 127  
last "Lovelace"

Look here!

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!

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level Function File Line Address Number Function File Line Address Condition

1 nameHash NameHash.cpp 62 0x555555b67cb 1 ...g) ...eHash.cpp 66 ...5555b6782

2 studentMain NameHash.cpp 31 0x555555b6595

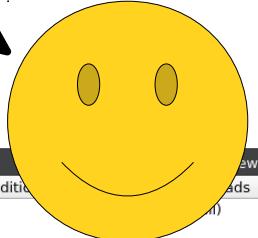
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent... 0x5555556161bc

4 GThreadStd::run() 0x555555f9476

5 ?? 0xfffff6143d84

6 start\_thread pthread\_create.c 463 0xfffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Activities    Qt Creator ▾

Jan 4 3:53 PM  
NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects  
NameHash [main]  
Sources  
NameHash.cpp

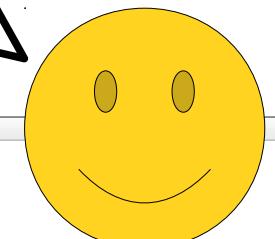
```

48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
52      * 2^31 - kLargePrime - 1.
53     */
54
55     for (char ch: first + last) {
56         /* Convert the input character to lower case
57          * lower-case letters are always less than
58          */
59         ch = tolower(ch);
60         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
61     }
62     return hashVal;
63 }
64
65 }
```

Line: 62, Col: 5

Name	Value	Type
__for_begin	@0x7ffc6058c78	std::string::iterator
__for_end	@0x7ffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash	NameHash.cpp	62	0x555555b67cb	1	...g)	...eHash.cpp	66	...5555b6782	(all)		
2	studentMain	NameHash.cpp	31	0x555555b6595								
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc								
4	GThreadStd::run()			0x555555f9476								
5	??			0x7ffff6143d84								
6	start_thread	pthread_create.c	463	0x7ffff6257590								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

Activities Qt Creator ▾ Jan 4 3:53 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .  
53 \*/  
54

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

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

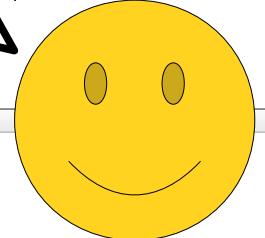
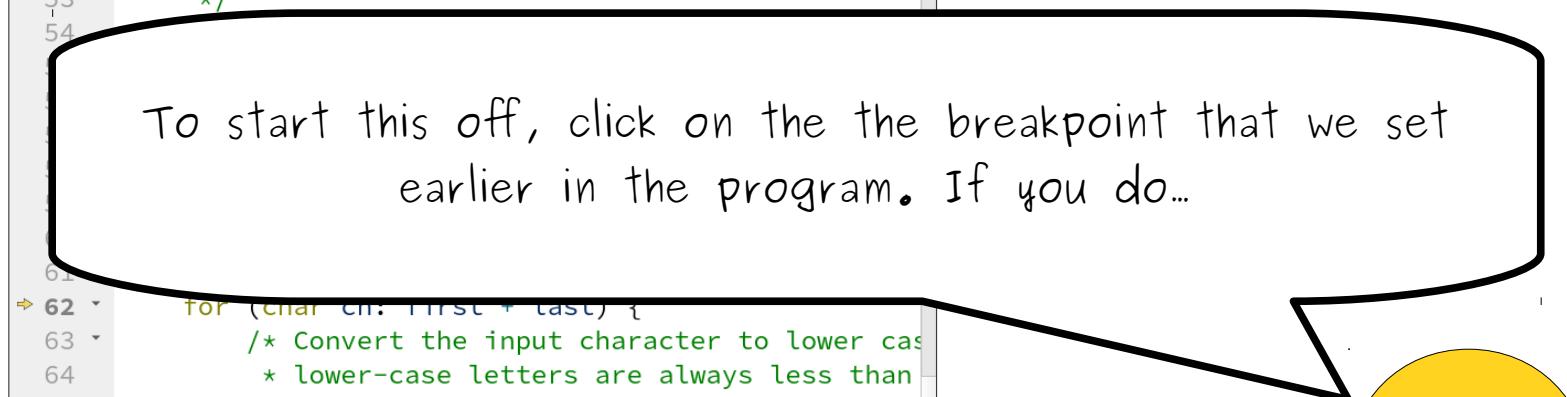
62

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

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

1 nameHash NameHash.cpp 62 0x555555b67cb 1 ...g) ...eHash.cpp 66 ...5555b6782  
2 studentMain NameHash.cpp 31 0x555555b6595  
3 std::Function\_handler<int ()>, QtGui::startBackgroundEvent  
4 GThreadStd::run()  
5 ??  
6 start thread pthread\_create.c 463 0x7ffff6143d84

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views (all)



Activities Qt Creator ▾

Jan 4 3:57 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

48 \*/  
49 int nameHash(string first, string last){  
50 /\* This hashing scheme needs two prime numbers,  
51 \* prime. These numbers were chosen because the  
52 \*  $2^{31} - kLargePrime - 1$ .  
53 \*/  
54

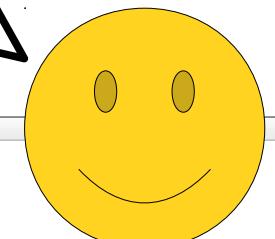
61  
62 for (char ch: first + last) {  
63 /\* Convert the input character to lower case  
64 \* lower-case letters are always less than  
65 \*/  
66 ch = tolower(ch);  
67 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
68 }  
69 return hashVal;  
70 }  
71

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'd' 100 char  
first "Ada" std::string  
hashVal ??? int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string

... 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!

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities Qt Creator ▾

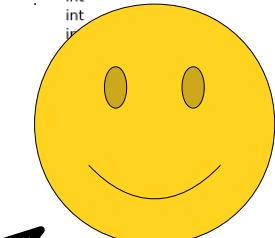
Jan 4 3:57 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch;
67         hashVal += ch;
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'd' char  
first "Ada" std::string  
hashVal ??? int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string



Now, take a look back at these buttons.

Debugger GDB for "NameHash" ▾

Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	62	0x5555555b67cb
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator() (const QThread*, void*)	QtGui::startBackgroundEventLoop	0x5555556161bc	0x5555555f9476
4	GThreadStd::run()	0xfffff6143d84	0x7ffff6143d84	pthread_create.c 463 0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Activities Qt Creator ▾

Jan 4 3:57 PM NameHash.cpp @ NameHash [main] - Qt Creator

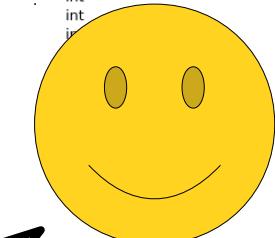
File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch;
67         hashVal = (hashVal * kSmallPrime + ch);
68     }
69     return hashVal;
70 }
71 }
```

Line: 62, Col: 5

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'd' char  
first "Ada" std::string  
hashVal ??? int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string



Hover your mouse over the one that's on the far right. When you hover over it, it should say "Step Out."

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results Views

Activities Qt Creator ▾

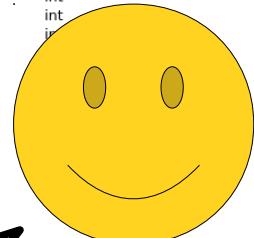
Jan 4 3:57 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch
67         has
68     }
69     return hashVal;
70 }
71 }
```

Name Value Type  
for\_begin @0x7ffc6058c78 std::string::iterator  
for\_end @0x7ffc6058c80 std::string::iterator  
for\_range "AdaLovelace" std::string &&  
ch 'd' char  
first "Ada" std::string  
hashVal ??? int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last "Lovelace" std::string



Don't click just yet. But when you do click,  
it will run the rest of the `nameHash`  
function until it finishes and returns.

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	62	0x5555555b67cb
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::Function_handler<int ()>, QtGui::startBackgroundEvent()			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread.c	463	0x7ffff6257590

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results Views

Activities Qt Creator ▾

Jan 4 3:57 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

```
48     */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers,
51      * prime. These numbers were chosen because the
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
60      * name, updating the hash at each step.
61      */
62     for (char ch : first) {
63         /*
64          *
65          */
66         ch;
67         hashVal = ...;
68     }
69     return hashVal;
70 }
71 }
```

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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results 8 Views

Table:

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	62	0x5555555b67cb
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::function<int ()>::operator()	0x5555556161bc		
4	GThreadStd::run()	0x5555555f9476		
5	??	0xfffff6143d84		
6	start_thread	pthread_create.c	463	0xfffff6257590

Activities Qt Creator ▾

Jan 4 4:02 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

NameHash.cpp <Select Symbol> Unix (LF) Line: 31, Col: 5

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * declare it in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash of your name is: " << hashValue;
32     return 0;
33 }
34
35 /* This is
   * to talk
   * the mean
   * of the i
   *
   * For thos
   * treats e
36
37 */

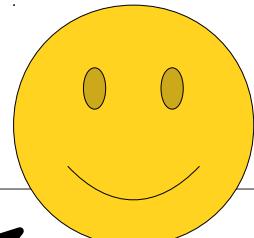
... you should end up with something that
looks like this!
```

returned value 1967457 int

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555b6595								
2	std::_Function_handler<int ()>, QWidget::startBackgroundEvent()			0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Activities Qt Creator ▾ Jan 4 4:02 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

NameHash.cpp

```
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us
   * declare it in main and then define it later in the program.
   */
23 int nameHash(string first, string last);
24
25 int main() {
26     string first = getLine("What is your first name?");
27     string last = getLine("What is your last name?");
28
29     int hashValue = nameHash(first, last);
30
31     cout << "The hash of your name is: " << hashValue;
32     return 0;
33 }
34
35 /* This is
   * to talk
   * the mean
   * of the i
   *
   * For thos
   * treats e
36
37 */

  
```

Line: 31, Col: 5

Name Value Type

Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

returned value 1967457 int

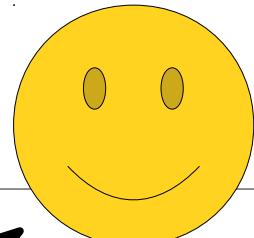
Let's take a minute to get our bearings.  
Where exactly are we?

Debugger GDB for "NameHash" Threads: #12 NameHash

Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Funct	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555b6595								
2	std::_Function_handler<int ()>, QWidget::startBackgroundEve...			0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Activities Qt Creator ▾

Jan 4 4:02 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

NameHash.cpp

#include "simpio.h" // for getLine  
using namespace std;  
  
/\* Prototype for the nameHash function. This lets us  
 \* in main and then define it later in the program.  
 \*/  
int nameHash(string first, string last);  
  
int main() {  
 string first = getLine("What is your first name?  
 string last = getLine("What is your last name?  
  
 int hashValue = nameHash(first, last);  
  
 cout << "The hash of your name is: " << hashValue  
 return 0;  
}  
  
/\* This is  
 \* to talk  
 \* the mean  
 \* of the i  
 \*  
 \* For those  
 \* treats e

Line: 31, Col: 5

Name Value Type  
first "Ada" std::string  
hashValue 0 int  
last "Lovelace" std::string

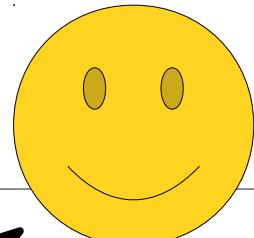
returned value 1967457 int

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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Funct	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555b6595								
2	std::_Function_handler<int ()>, QWidget::startBackgroundEve...			0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Activities Qt Creator ▾

Jan 4 4:02 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

NameHash.cpp

#include "simpio.h" // for getLine  
using namespace std;  
  
/\* Prototype for the nameHash function. This lets us  
 \* in main and then define it later in the program.  
 \*/  
int nameHash(string first, string last);  
  
int main()  
{  
 string first = getLine("What is your first name?  
 string last = getLine("What is your last name?  
  
 int hashValue = nameHash(first, last);  
  
 cout << "The hash of your name is: " << hashValue  
 return 0;  
}  
  
/\* This is the  
 \* to talk m  
 \* the meant  
 \* of the in  
 \*  
 \* For those  
 \* treats ea

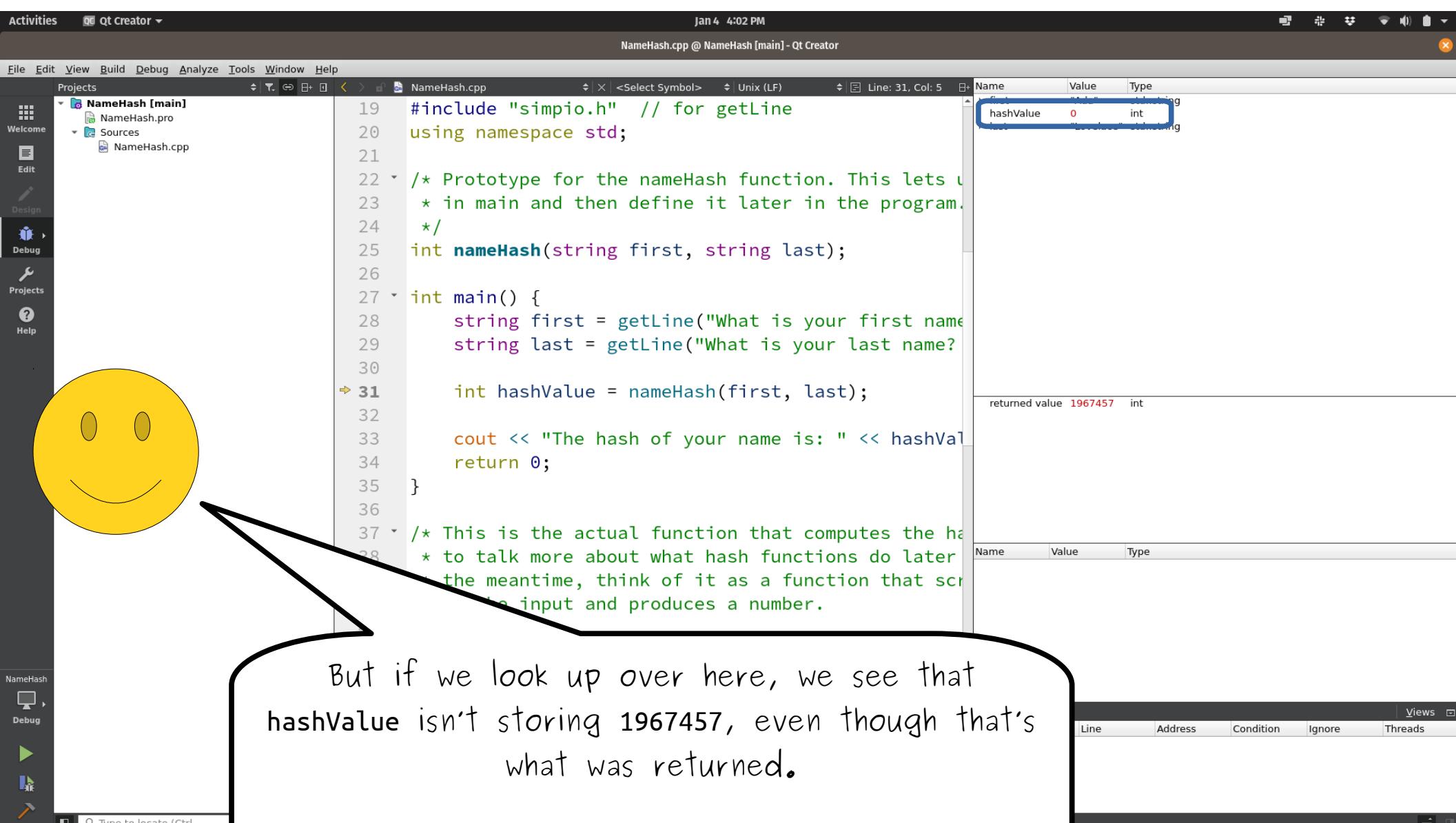
Debugger GDB for "NameHash"  
Level Function  
1 studentMain  
2 std::function\_handler<int ()>, QtG  
3 GThreadStd::run()  
4 ??  
5 start\_thread  
6 clone

Line: 31, Col: 5

Name Value Type  
first "Ada" std::string  
hashValue 0 int  
last "Lovelace" std::string

returned value 1967457 int

We can see that the nameHash function returned 1967457. Thanks, debugger!  
(A note: it seems like on some Macs, this number doesn't display. Don't worry if you don't see it - just continue on as usual.)



A screenshot of the Qt Creator IDE. The main window shows a code editor with the file `NameHash.cpp` open. The code defines a function `nameHash` that takes two strings and returns an integer hash value. A callout bubble from a smiley face icon points to the variable `hashValue` in the code editor, highlighting the line `int hashValue = nameHash(first, last);`. In the bottom right corner of the code editor, there is a status message: "returned value 1967457 int". To the right of the code editor is a debugger's registers view, which shows a table with one row:

Name	Value	Type
hashValue	0	int

The status bar at the bottom of the IDE shows the path `NameHash`, the current file `Debug`, and other system information.

But if we look up over here, we see that `hashValue` isn't storing 1967457, even though that's what was returned.

(You might see a number other than 0 on your system - that's okay.)



Jan 4 4:02 PM

NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

#include "simpio.h"  
using namespace std;  
  
/\* Prototype for nameHash  
 \* in main and here  
int nameHash(string first, string last);  
  
int main() {  
 string first = getLine("What is your first name?");  
 string last = getLine("What is your last name?");  
  
 int hashValue = nameHash(first, last);  
  
 cout << "The hash of your name is: " << hashValue;  
 return 0;  
}  
  
/\* This is the actual function that computes the hash.  
 \* To talk more about what hash functions do later,  
 \* the meantime, think of it as a function that scans  
 \* of the input and produces a number.  
 \*  
 \* For those of you who are more mathematically inclined,  
 \* treats each character in the input name as a number  
 \* and adds them together.  
 \*/

Line: 31, Col: 5

Name Value Type  
first "Ada" std::string

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

returned value 1967457 int

Views

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555b6595								
2	std::__function_handler<int ()>	QtGui::startBackgroundEventLoop()		0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Jan 4 4:02 PM

NameHash.cpp @ NameHash [main] - Qt Creator

This is pretty cool, actually!

```
File Edit View Build Debug Analyze Tools Window Help
Projects Sources NameHash.cpp
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for nameHash()
23 * in main and here
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;
34     return 0;
35 }
36
37 /* This is the actual function that computes the hash.
38 * To talk more about what hash functions do later,
39 * the meantime, think of it as a function that scrunches
40 * all the letters of the input and produces a number.
41 *
42 * For those of you who are more mathematically inclined,
43 * treats each character in the input name as a number
44 * and adds them up. The result is the hash value.
45 */
46
47 int nameHash(string first, string last) {
48     int hashValue = 0;
49     for (int i = 0; i < first.length(); i++) {
50         hashValue += first[i];
51     }
52     for (int i = 0; i < last.length(); i++) {
53         hashValue += last[i];
54     }
55     return hashValue;
56 }
```

returned value 1967457 int

Name	Value	Type
first	"Ada"	std::string

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555b6595								
2	std::__function_handler<int ()>::QtGui::startBackgroundEvent()			0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results



Jan 4 4:02 PM

NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

```
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for nameHash()
23 * in main and here
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;
34     return 0;
35 }
36
37 /* This is the actual function that computes the hash
38 * to talk more about what hash functions do later
39 * the meantime, think of it as a function that scans
40 * of the input and produces a number.
41 *
42 * For those of you who are more mathematically inclined,
43 * treats each character in the input name as a number
44 * and adds them up to produce the hash value.
45 */
46
47 // nameHash.cpp
```

What's happened is that we've called the function from nameHash with a value. We're going through the program, but we haven't actually assigned anything to hashValue yet.

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!

returned value: 1967457 int

Name	Value	Type
------	-------	------

Debugger ♦ GDB for "NameHash" ♦				Threads: ➔ #12 NameHash	♦	Stopped: "function-finished"
Level	Function	File	Line	Address	Number	Funct File
1	studentMain	NameHash.cpp	31	0x5555555b6595		
2	std::Function_handler<int ()>::QtGui::startBackgroundEve...			0x55555556161bc		
3	GThreadStd::run()			0x5555555f9476		
4	??			0x7ffff6143d84		
5	start_thread	pthread_create.c	463	0xfffff6257590		
	else	else.c	25	0xfffff5e32232		

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 OML Debugger Console 6 Clone 7 Version Control 8 Test Results



Jan 4 4:02 PM

NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

#include "simpio.h"  
using namespace std;  
/\* Prototype for  
 \* in main and  
 \*/  
int nameHash(string first, string last);  
  
int main() {  
 string first = getLine("What is your first name?");  
 string last = getLine("What is your last name?");  
  
 int hashValue = nameHash(first, last);  
  
 cout << "The hash of your name is: " << hashValue;  
 return 0;  
}  
  
/\* This is the actual function that computes the hash.  
 \* To talk more about what hash functions do later,  
 \* the meantime, think of it as a function that scans  
 \* of the input and produces a number.  
 \*  
 \* For those of you who are more mathematically inclined,  
 \* treats each character in the input name as a number  
 \* and adds them together to produce the hash value.  
 \*/

Line: 31, Col: 5

Name Value Type  
first "Ada" std::string

returned value 1967457 int

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "function-finished".

Level Function File Line Address

1 studentMain NameHash.cpp 31 0x5555555b6595  
2 std::\_\_function\_handler<int ()>::operator()() const@Eve... 0x5555556161bc  
3 GThreadStd::run() 0x5555555f9476  
4 ?? 0x7ffff6143d84  
5 start\_thread pthread\_create.c 463 0x7ffff6257590  
6 clone clone.S 95 0x7ffff5e30223

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

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



Activities Qt Creator ▾

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

Jan 4 4:07 PM NameHash.cpp @ NameHash [main] - Qt Creator

21  
22 /\* Prototype for the nameHash function. This lets us  
23 \* in main and then define it later in the program.  
24 \*/  
25 int nameHash(s  
26  
27 int main() {  
28 string fir  
29 string las  
30  
31 int hashValue = nameHash(first, last);  
32  
33 cout << "The hash of your name is: " << hashValue  
34 return 0;  
35 }  
36  
37 /\* This is the actual function that computes the ha  
38 \* to talk more about what hash functions do later  
39 \* the meantime, think of it as a function that scr  
40 \* of the input and produces a number.  
41 \*  
42 \* For those of you who are more mathematically in  
43 \* treats each character in the input name as a num  
44 \* It then uses them as coefficients in a polynomia  
45 \* F\_p, where p is a large prime number, and evalua

Line: 33, Col: 5

Name Value Type

first "Ada" std::string  
hashValue 1967457 int  
last "Lovelace" std::string

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

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	33	0x5555555b65b3								
2	std::__function_handler<int ()>, QWidget::startBackgroundEve...			0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results



Jan 4 4:07 PM

NameHash.cpp @ NameHash [main] - Qt Creator

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

```
File Edit View Build Debug Analyze Tools Window Help
Projects NameHash [main]
  Sources NameHash.pro
  Sources NameHash.cpp
Line: 33, Col: 5
Name Value Type
first "Ada" std::string
```

```
21
22  /* Prototype for nameHash()
23  * in main and nameHash()
24  */
25 int nameHash(string first, string last) {
26
27  int main() {
28      string first = "Ada";
29      string last = "Lovelace";
30
31      int hashValue = nameHash(first, last);
32
33      cout << "The hash of your name is: " << hashValue;
34      return 0;
35  }
36
37  /* This is the actual function that computes the hash.
38  * To talk more about what hash functions do later,
39  * the meantime, think of it as a function that scrunches
40  * of the input and produces a number.
41  *
42  * For those of you who are more mathematically inclined,
43  * treats each character in the input name as a number.
44  * It then uses them as coefficients in a polynomial
45  * F_p, where p is a large prime number, and evaluates
  
```

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	studentMain	NameHash.cpp	33	0x5555555b65b3								
2	std::__function_handler<int ()>::QtGui::startBackgroundEvent(QEvent*)			0x5555556161bc								
3	GThreadStd::run()			0x5555555f9476								
4	??			0x7ffff6143d84								
5	start_thread	pthread_create.c	463	0x7ffff6257590								
6	clone	clone.S	95	0x7ffff5e30223								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 Version Control 7 Test Results

Activities Qt Creator ▾

Jan 4 4:07 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

21  
22 */\* Prototype for the nameHash function. This lets us...\*/*  
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;  
34     return 0;  
35 }  
36  
37 */\* This is the actual function that computes the hash value. We'll...\*/*  
38   *\* to talk more about what hash functions do later.*  
39   *\* the meantime, think of it as a function that takes two strings...\*/*  
40   *\* of the input and produces one integer value.*

Name Value Type  
first Ada std::string  
hashValue 1967457 int  
last Lovelace std::string

To do this, click on this button. If you hover over it, it says "Continue," and that button means "unpause the program and let it keep running from here."

inc num combia valua

Stopped: "end-stepping-range".

Views

3 Onreadable (num)  
4 ??  
5 start\_thread  
6 clone

0x333333519476  
0xfffff6143d84  
0xfffff6257590  
clone.S 463 0x7ffffe30223

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

A yellow smiley face icon is overlaid on the bottom right of the code editor area. A hand-drawn callout bubble points from the text above to the 'Run' button in the toolbar at the bottom left. A mouse cursor arrow points towards the 'Run' button.

Activities NameHash ▾ Jan 4 4:08 PM NameHash.cpp @ NameHash [main] - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects NameHash [main] Sources NameHash.cpp

21  
22 */\* Prototype for the nameHash function. This lets us...\*/*  
23  
24  
25  
26 NameHash Console [Completed]  
27 File Edit Options Help  
28 What is your first name? Ada  
29 What is your last name? Lovelace  
30 The hash of your name is: 1967457  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40

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!

Debug Address Condition Ignore Threads

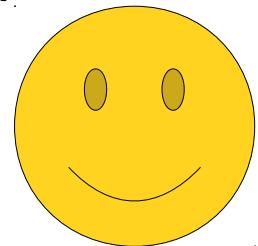
Type

1 student.h 0x5555556161bc  
2 std::\_Function\_handler<int ()>, QtGui::startBackgroundEvent... 0x5555555f9476  
3 GThreadStd::run() 0x7ffff6143d84  
4 ??  
5 start\_thread 0x7ffff6257590  
6 clone pthread\_create.c 463 0x7ffff5e30223  
clone.S 95 0x7ffff5e30223

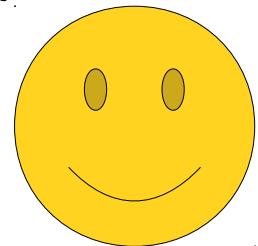
Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

<img alt="A yellow smiley face icon with a black outline, positioned next to the explanatory text." data-bbox="800 500 930 650>

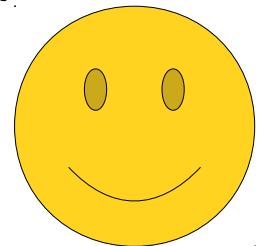
So there you have it! You've now gotten more  
familiar with the debugger!



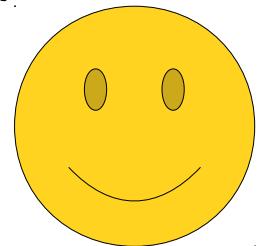
You know how to set a breakpoint to pause the program at a particular point.



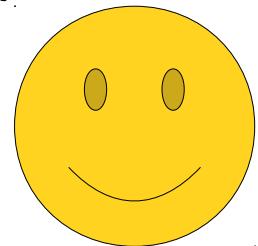
You know how to read the call stack and to see the values of local variables.



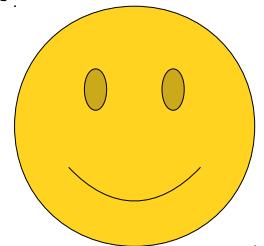
You know how to single-step the program and  
see what values change.



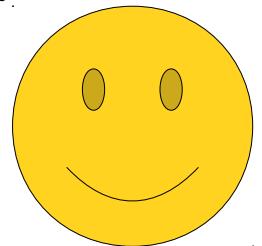
You know how to run a function to completion,  
and how to let the program keep on running.



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!

