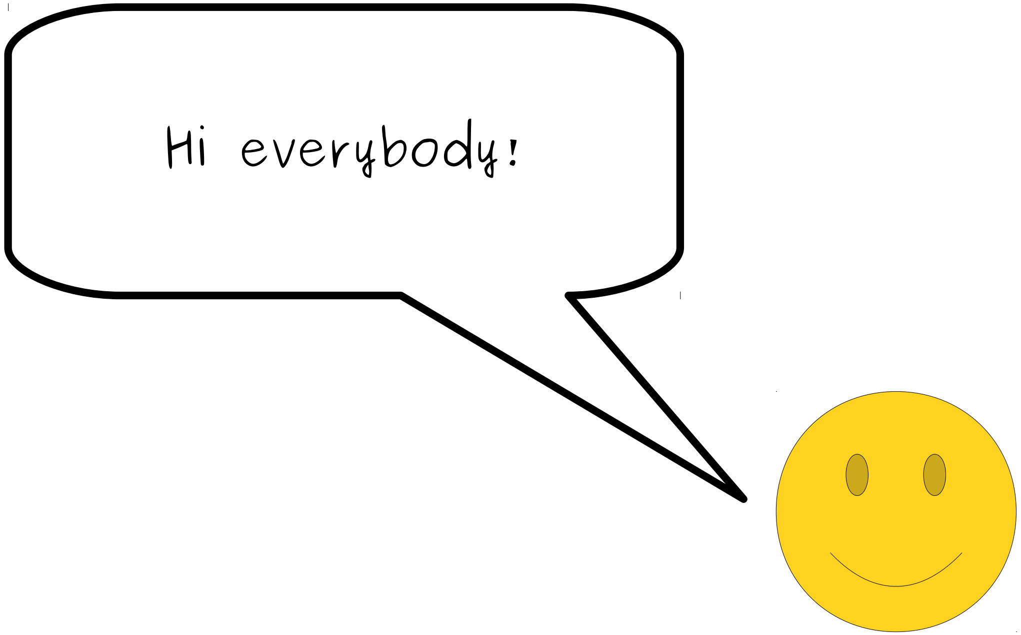
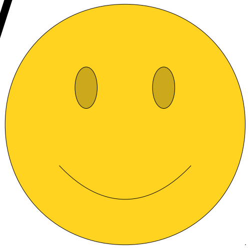
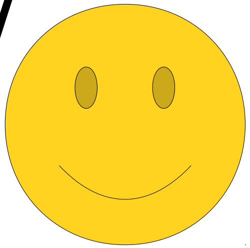
Assignment 0: Using the Debugger


Hi everybody!

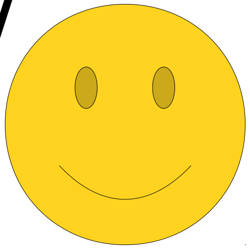
As part of Assignment 0, we'd like you to get a little bit of practice using the debugger in Qt Creator.



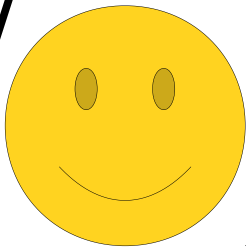
The debugger is a tool you can use to help see what your program is doing as you run it.



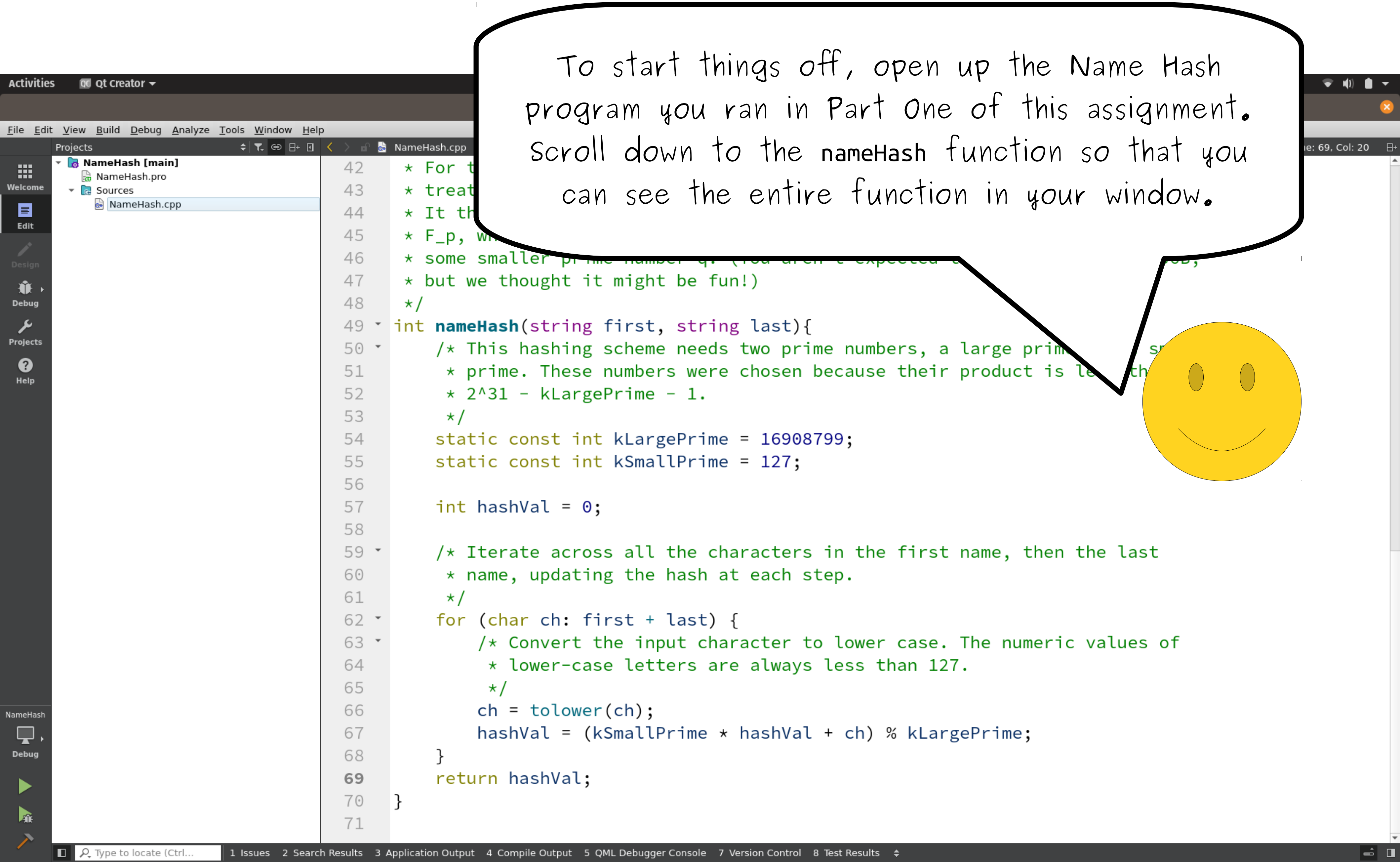
It's really useful for helping find errors in your programs, and the more practice you get with it, the easier it'll be to correct mistakes in the programs you write.



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.

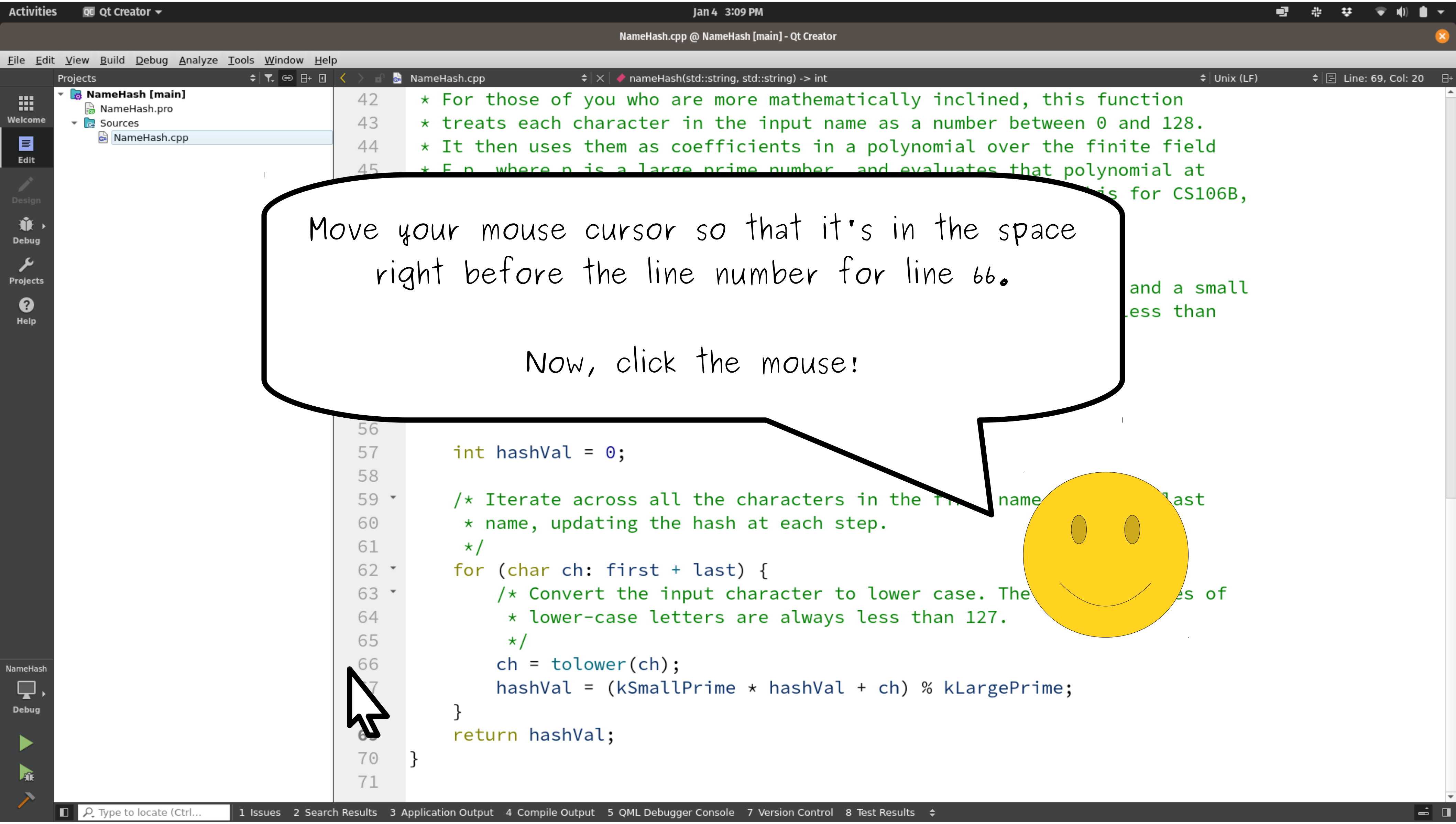


To start things off, open up the Name Hash program you ran in Part One of this assignment. Scroll down to the **nameHash** function so that you can see the entire function in your window.



Activities
Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
ne: 69, Col: 20
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

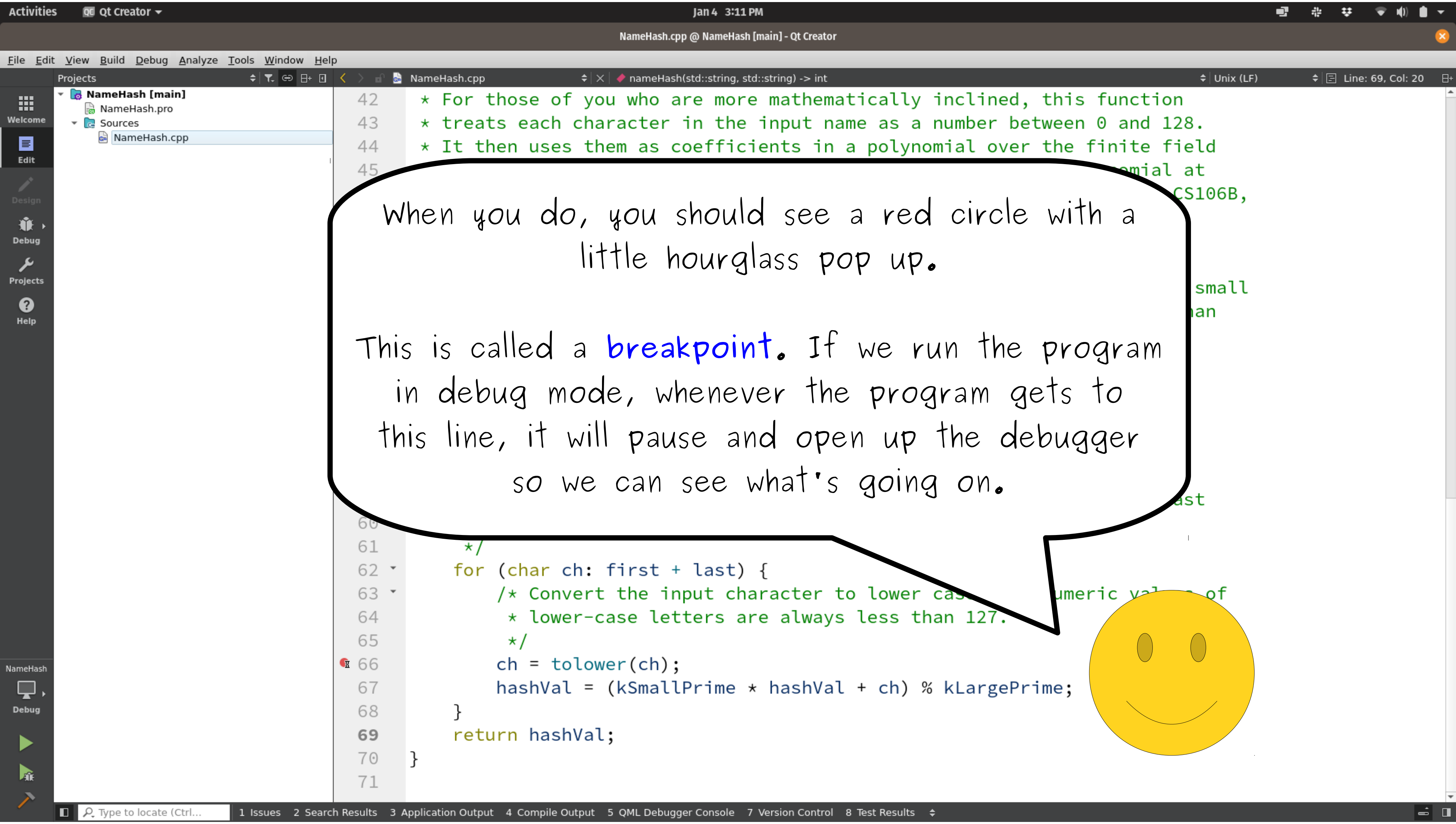
```
 * For t
 * treat
 * It th
 * F_p, wh
 * some smaller pr                                                        od,
 * but we thought it might be fun!)
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers, a large prim       sm
     * prime. These numbers were chosen because their product is le    th
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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



```
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
                                                            is for CS106B,

                                                            and a small
                                                            less than

56
57      int hashVal = 0;
58
59      /* Iterate across all the characters in the fi          name        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          es 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
```

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

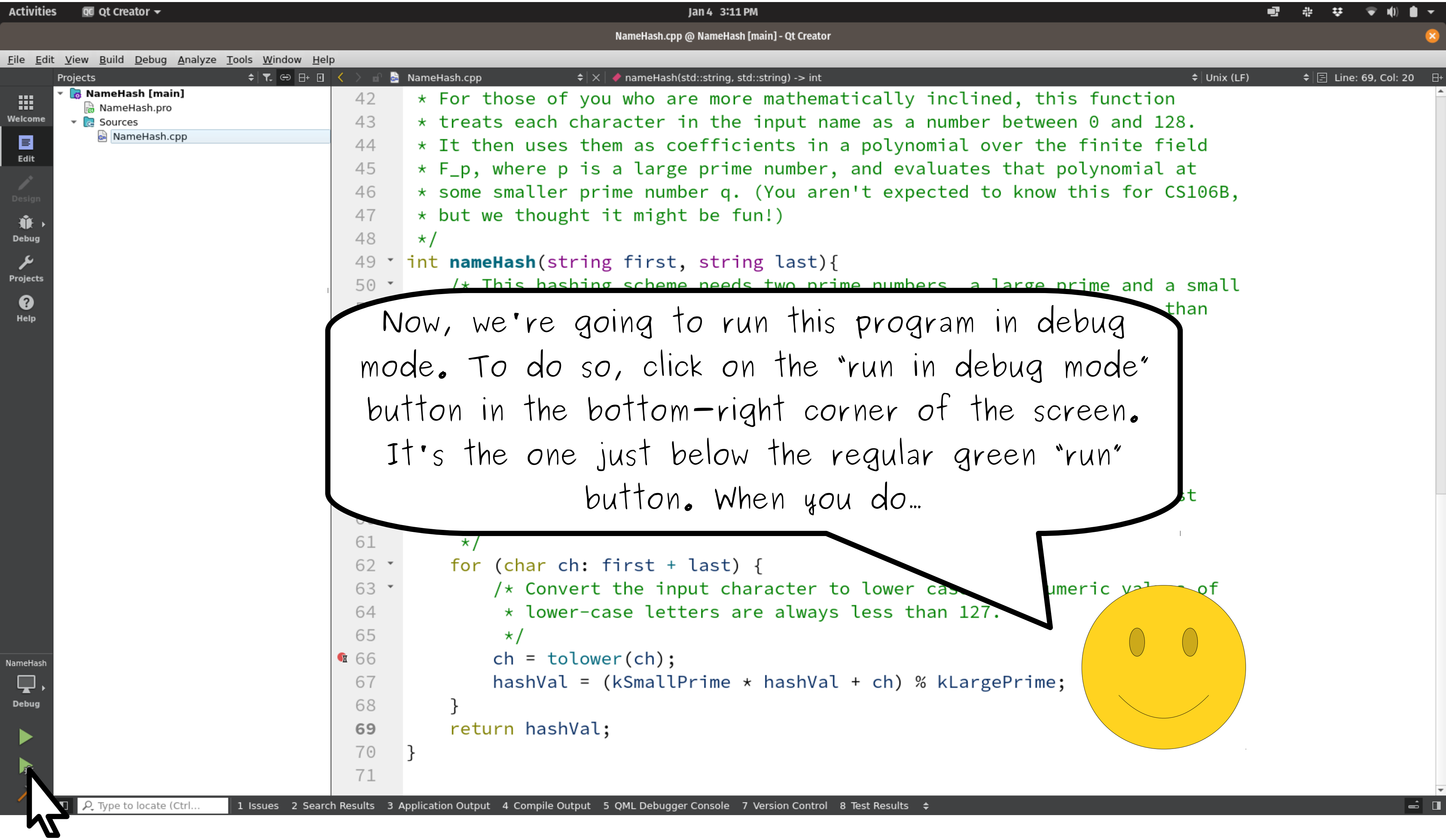
Now, click the mouse!


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]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
NameHash.cpp
nameHash(std::string, std::string) -> int
Unix (LF)
Line: 69, Col: 20
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

```
 * For those of you who are more mathematically inclined, this function
 * treats each character in the input name as a number between 0 and 128.
 * It then uses them as coefficients in a polynomial over the finite field
                                                       omial at
                                                       CS106B,

                                                        small
                                                        an

                                                        ast

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

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

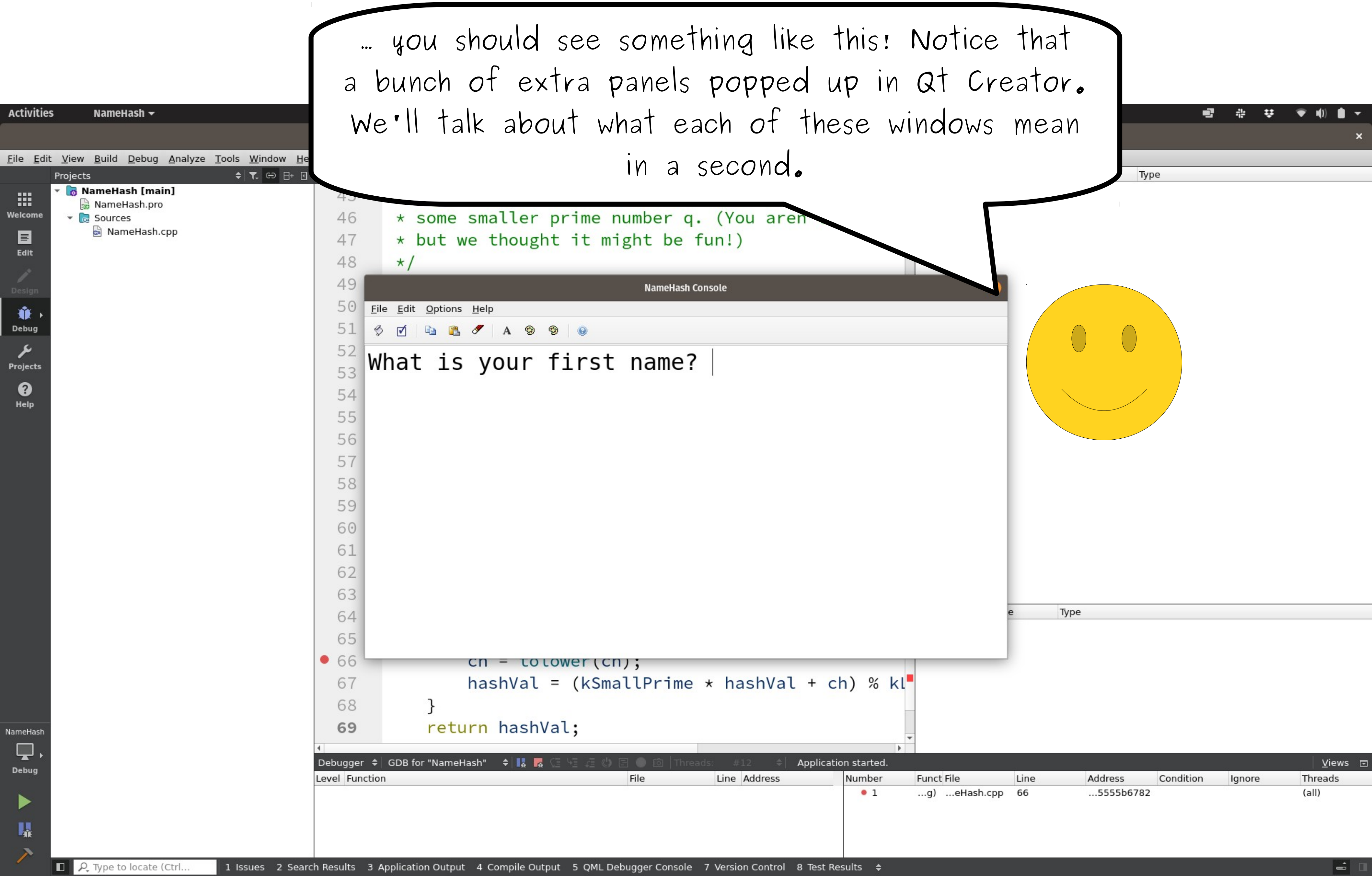
This is called a **breakpoint**. If we run the program in debug mode, whenever the program gets to this line, it will pause and open up the debugger so we can see what's going on.


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]
NameHash.pro
Sources
NameHash.cpp
nameHash(std::string, std::string) -> int
Unix (LF)
Line: 69, Col: 20
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
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

```
 * For those of you who are more mathematically inclined, this function
 * treats each character in the input name as a number between 0 and 128.
 * It then uses them as coefficients in a polynomial over the finite field
 * F_p, where p is a large prime number, and evaluates that polynomial at
 * some smaller prime number q. (You aren't expected to know this for CS106B,
 * but we thought it might be fun!)
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers, a large prime and a small
                                                                      than

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

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

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



```
 * some smaller prime number q. (You aren
 * but we thought it might be fun!)
 */
```

NameHash Console

```
What is your first name?
```

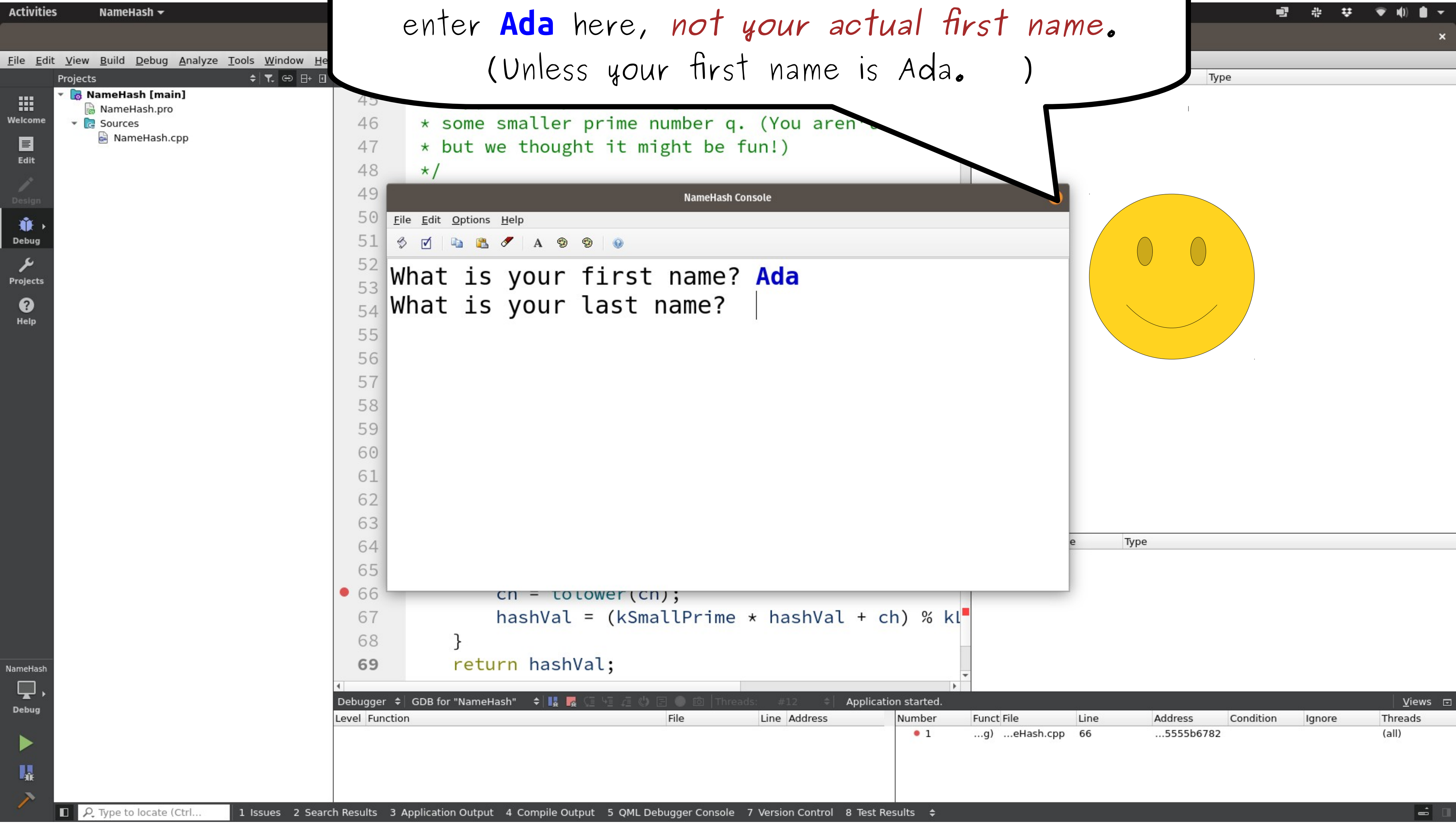
```
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
```

Application started.

Level	Function	File	Line	Address

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

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.)



```
 * some smaller prime number q. (You aren't
 * but we thought it might be fun!)
 */
```

NameHash Console

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

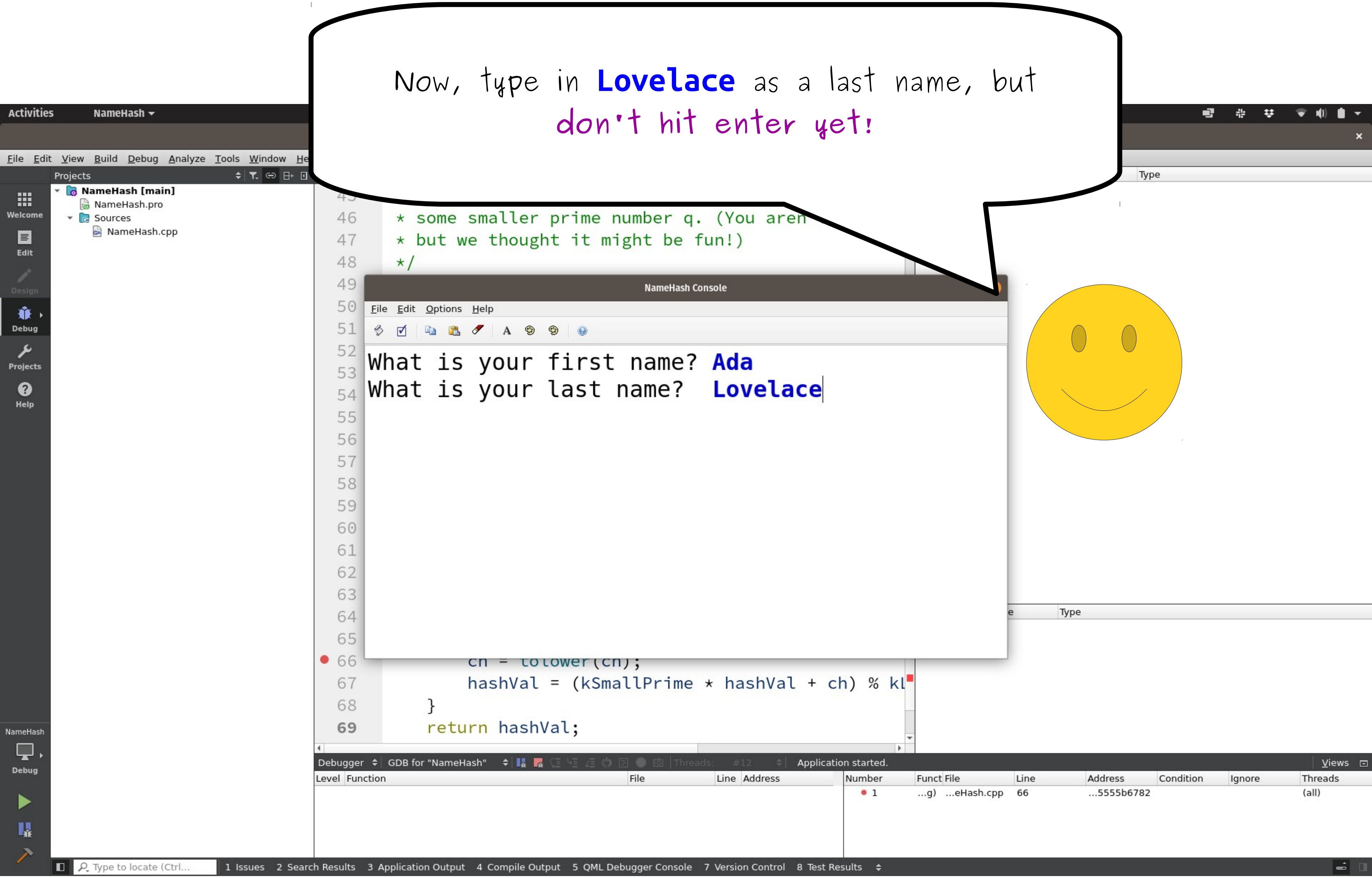
```
            ch = tolower(ch);
            hashVal = (kSmallPrime * hashVal + ch) % kL
        }
        return hashVal;
```

Application started.

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

Activities
NameHash
File Edit View Build Debug Analyze Tools Window He
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
File Edit Options Help
Type
Debugger
GDB for "NameHash"
Threads: #12
Views
Level Function File Line Address
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results
Type to locate (Ctrl...

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



```
 * some smaller prime number q. (You aren
 * but we thought it might be fun!)
 */
```

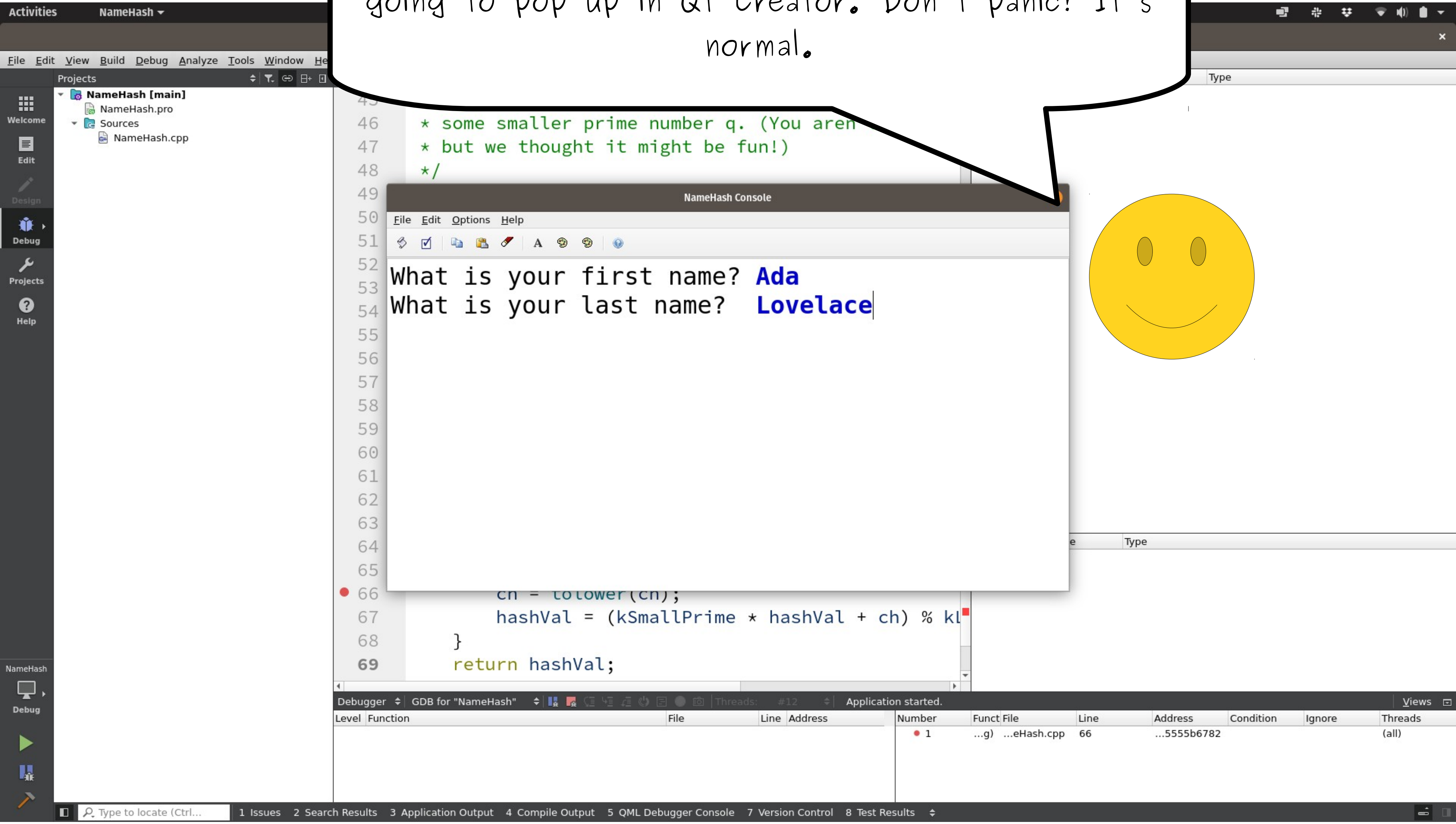
```
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
```

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

Application started.

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

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



```
 * some smaller prime number q. (You aren
 * but we thought it might be fun!)
 */
```

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

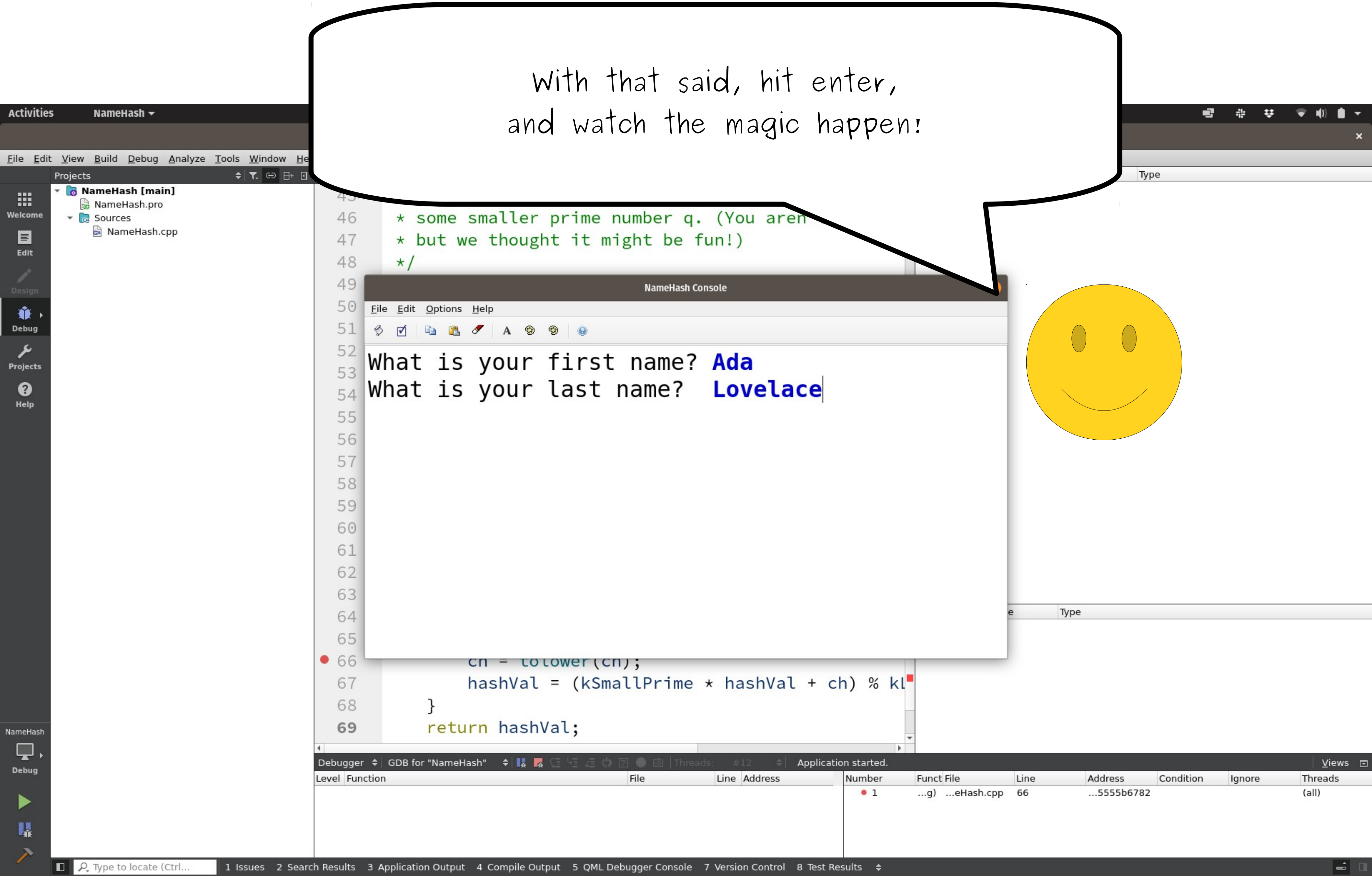
```
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
```

Application started.

Level	Function	File	Line	Address

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

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



```
 * some smaller prime number q. (You aren
 * but we thought it might be fun!)
 */
```

```
            ch = tolower(ch);
            hashVal = (kSmallPrime * hashVal + ch) % kL
        }
        return hashVal;
```

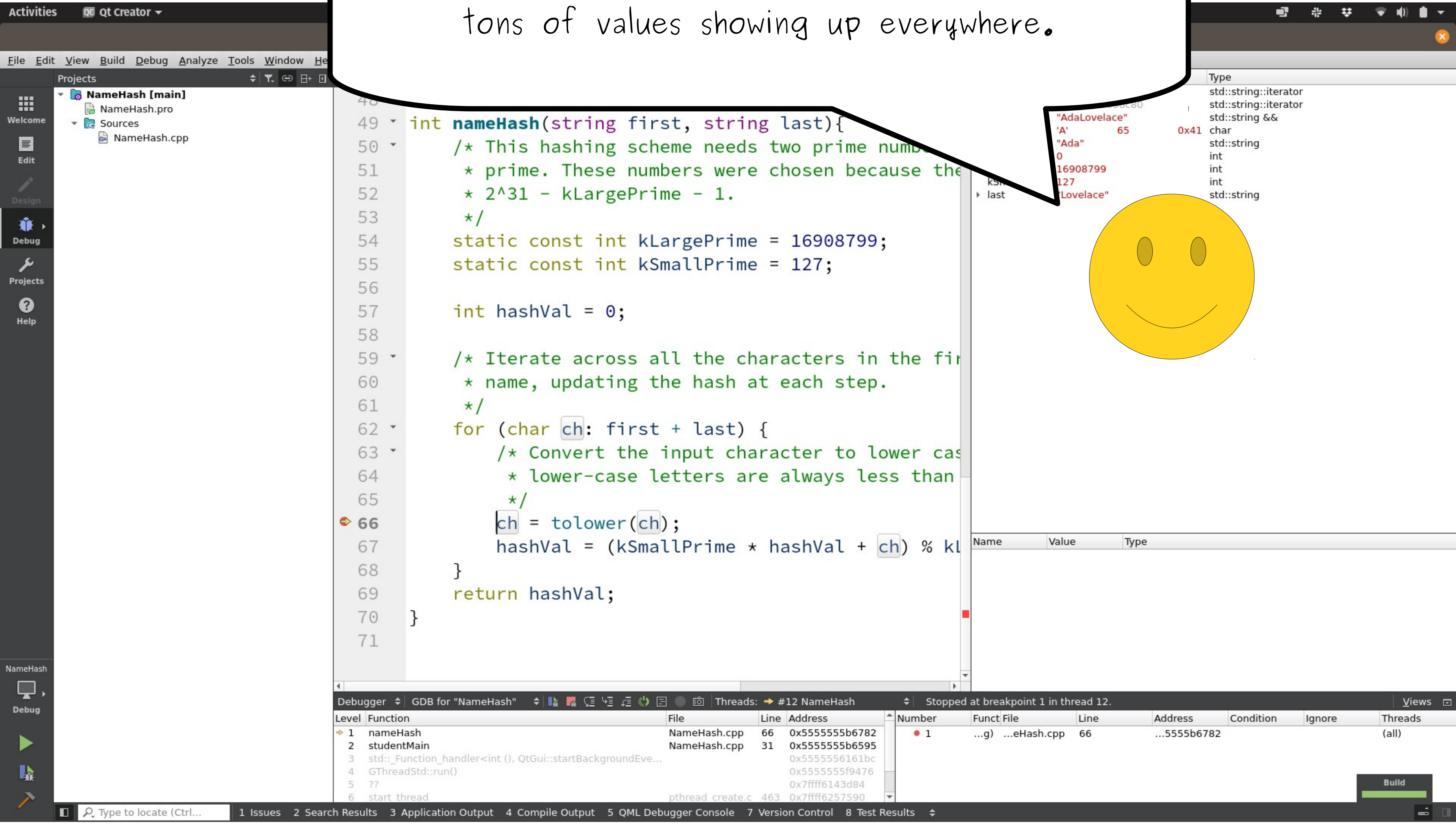
```
What is your first name? Ada
What is your last name?  Lovelace
```

Application started.

Level	Function	File	Line	Address

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

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



```
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numb
     * 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 fir
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

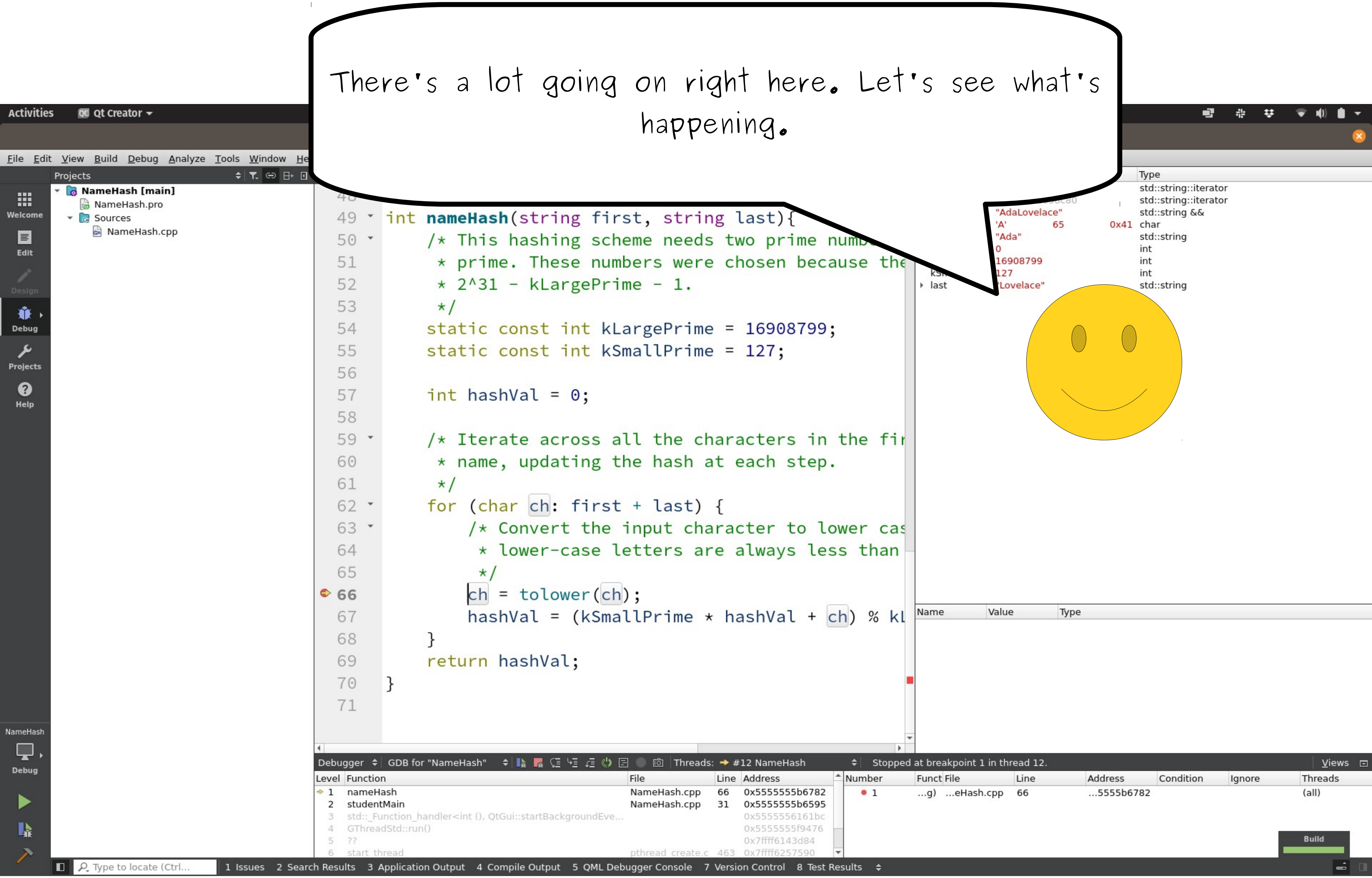
				Type
				std::string::iterator
				std::string::iterator
	"AdaLovelace"			std::string &&
	'A'	65	0x41	char
	"Ada"			std::string
	0			int
	16908799			int
kS	127			int
last	"Lovelace"			std::string

Name	Value	Type

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

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

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



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

```
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numb
     * 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 fir
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

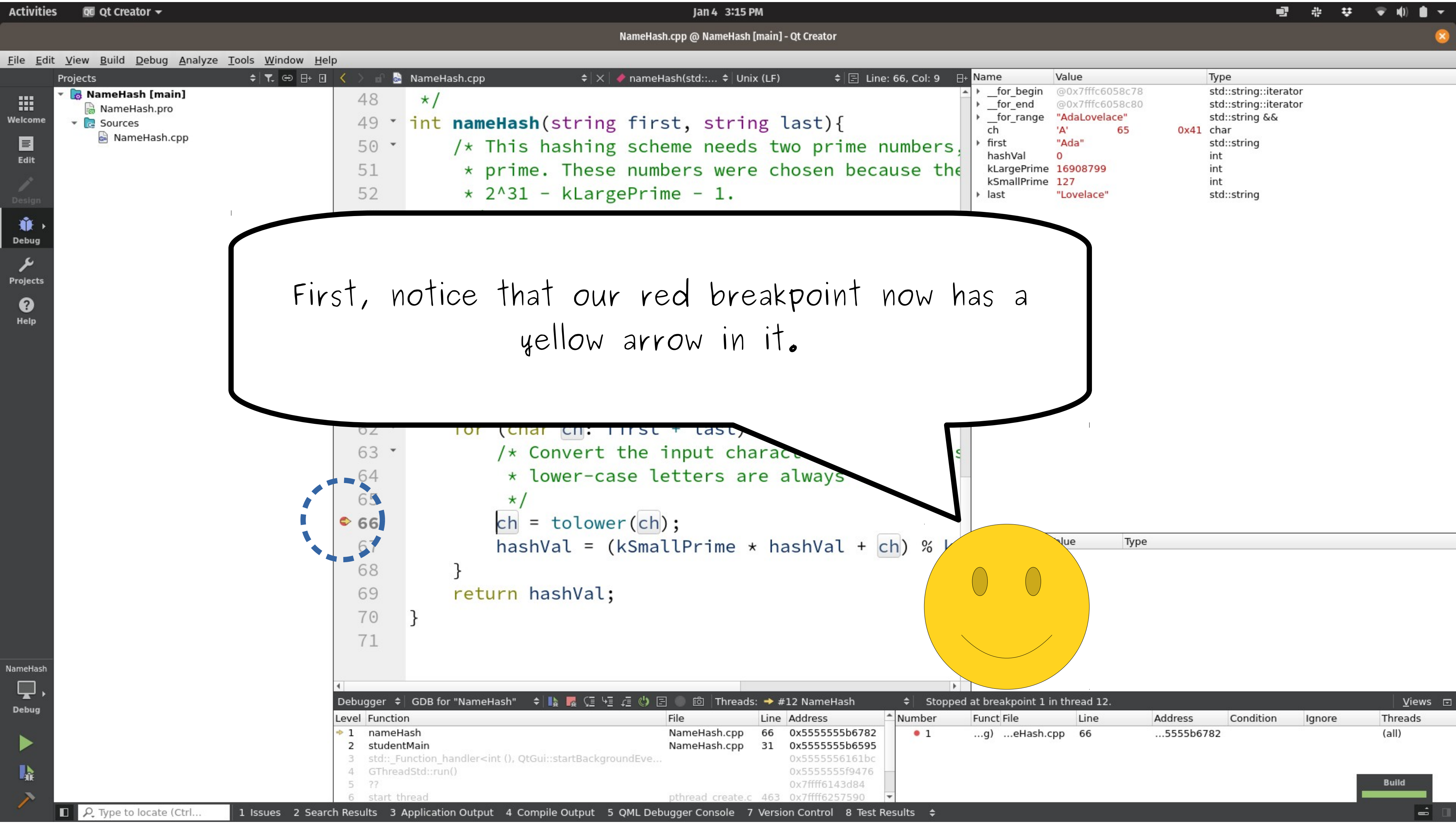
			Type
			std::string::iterator
			std::string::iterator
	"AdaLovelace"		std::string &&
	'A'	65 0x41	char
	"Ada"		std::string
	0		int
	16908799		int
kSr	127		int
last	"Lovelace"		std::string

Name	Value	Type

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

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

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



```
   */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
    for (char ch: first + last)
        /* Convert the input charac
         * lower-case letters are always
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % k
    }
    return hashVal;
}
```

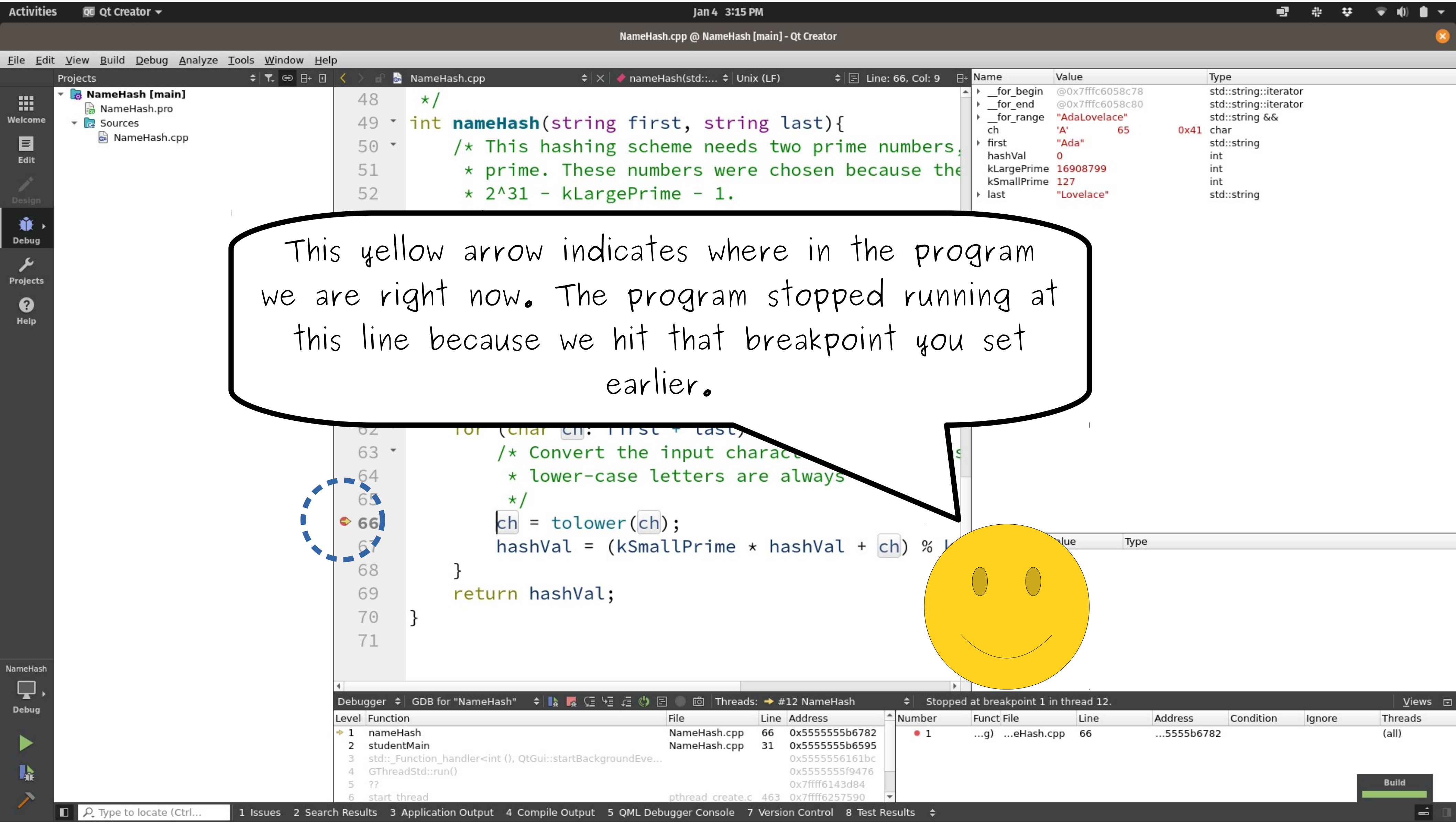
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Stopped at breakpoint 1 in thread 12.

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

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

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



```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
    for (char ch: first + last)
        /* Convert the input charac
         * lower-case letters are always
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % k
    }
    return hashVal;
}
```

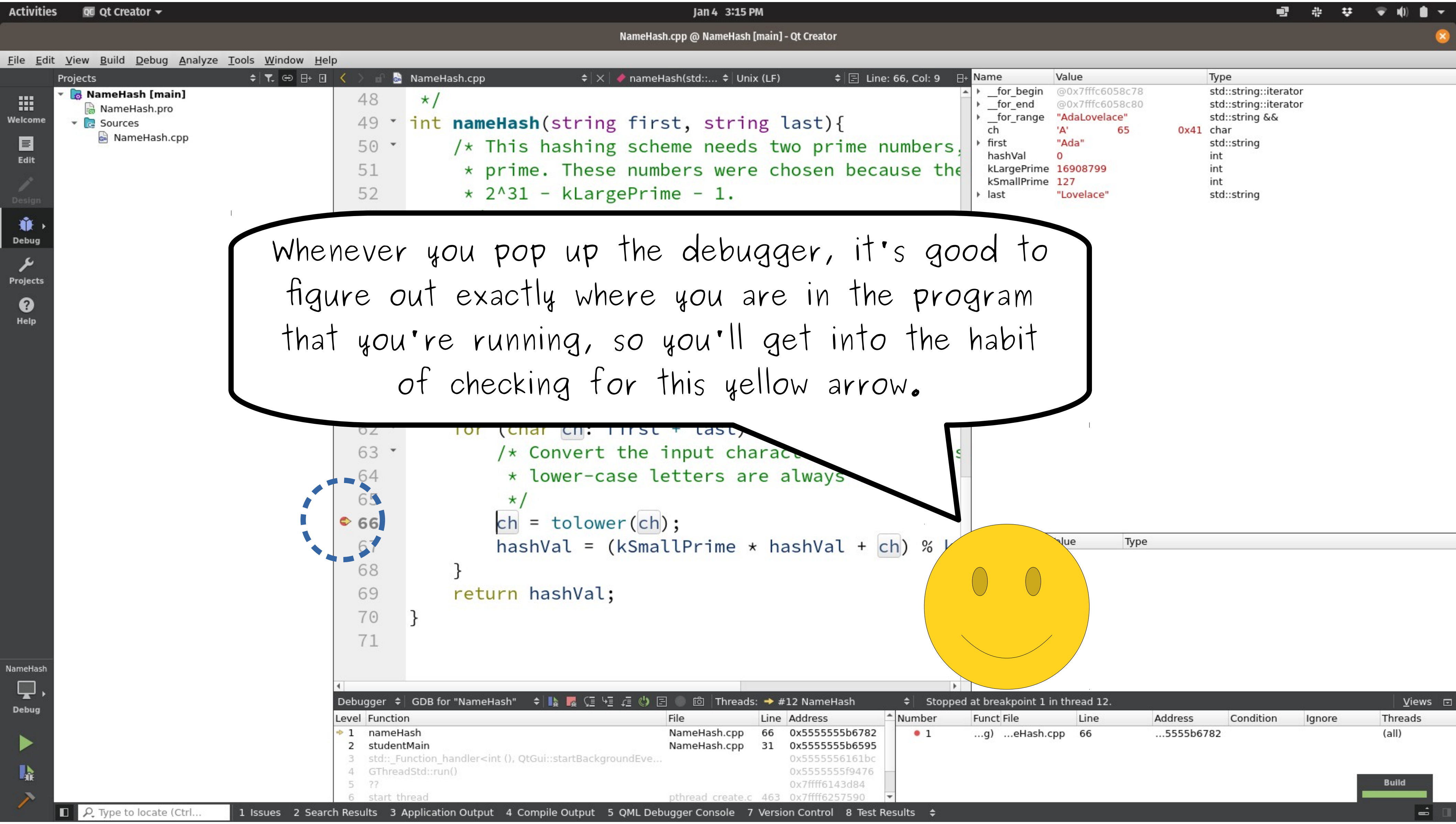
Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65 0x41	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Stopped at breakpoint 1 in thread 12.

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

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

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.


Activities
Qt Creator
Jan 4 3:15 PM
File Edit View Build Debug Analyze Tools Window Help
Projects
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Build
Views
Threads: #12 NameHash
Debugger
GDB for "NameHash"
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

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.

NameHash.cpp @ NameHash [main] - Qt Creator

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

NameHash.cpp | nameHash(std::... | Unix (LF) | Line: 66, Col: 9

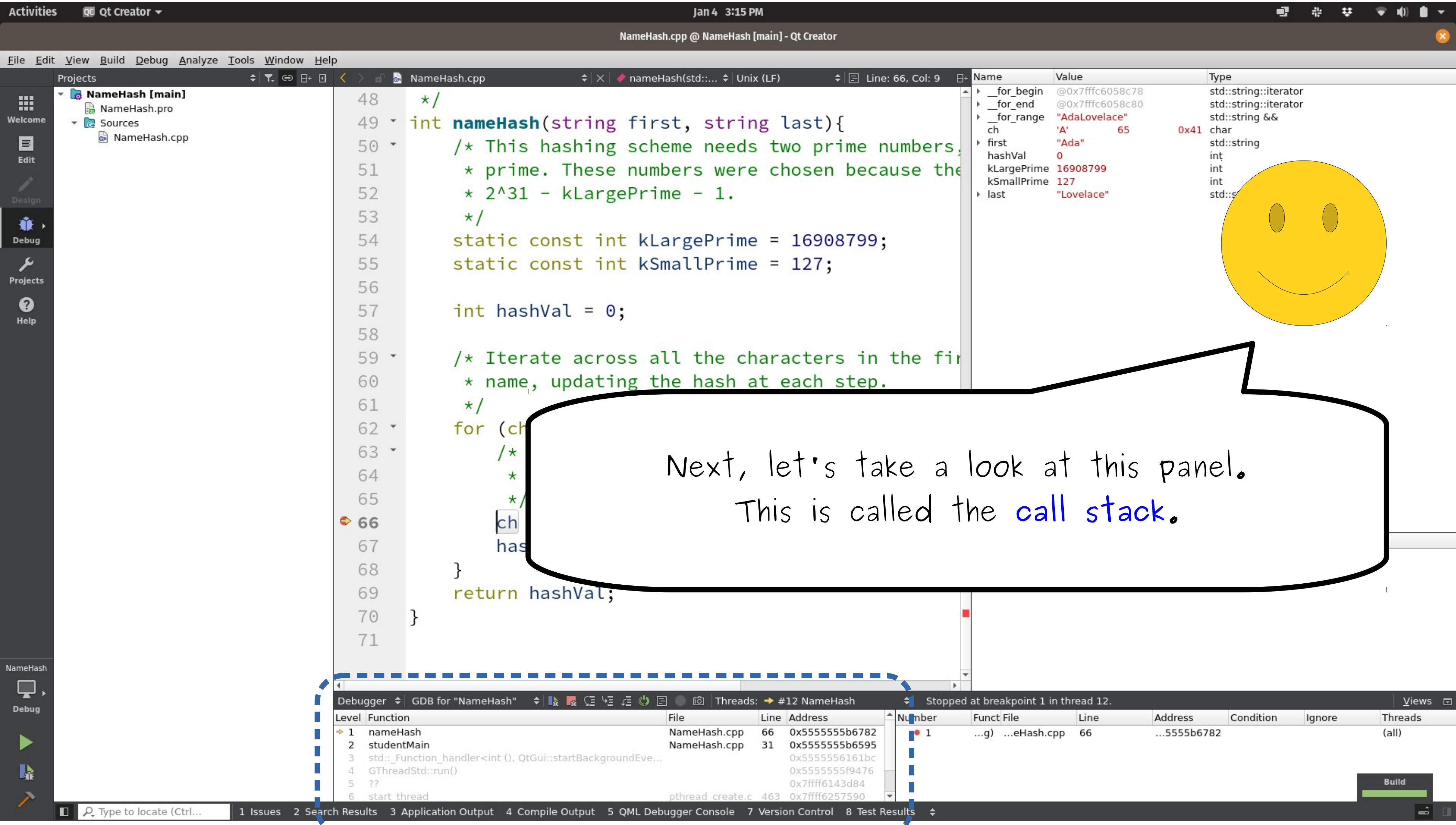
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
    for (char ch: first + last)
        /* Convert the input charac
         * lower-case letters are always
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % k
    }
    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	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
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



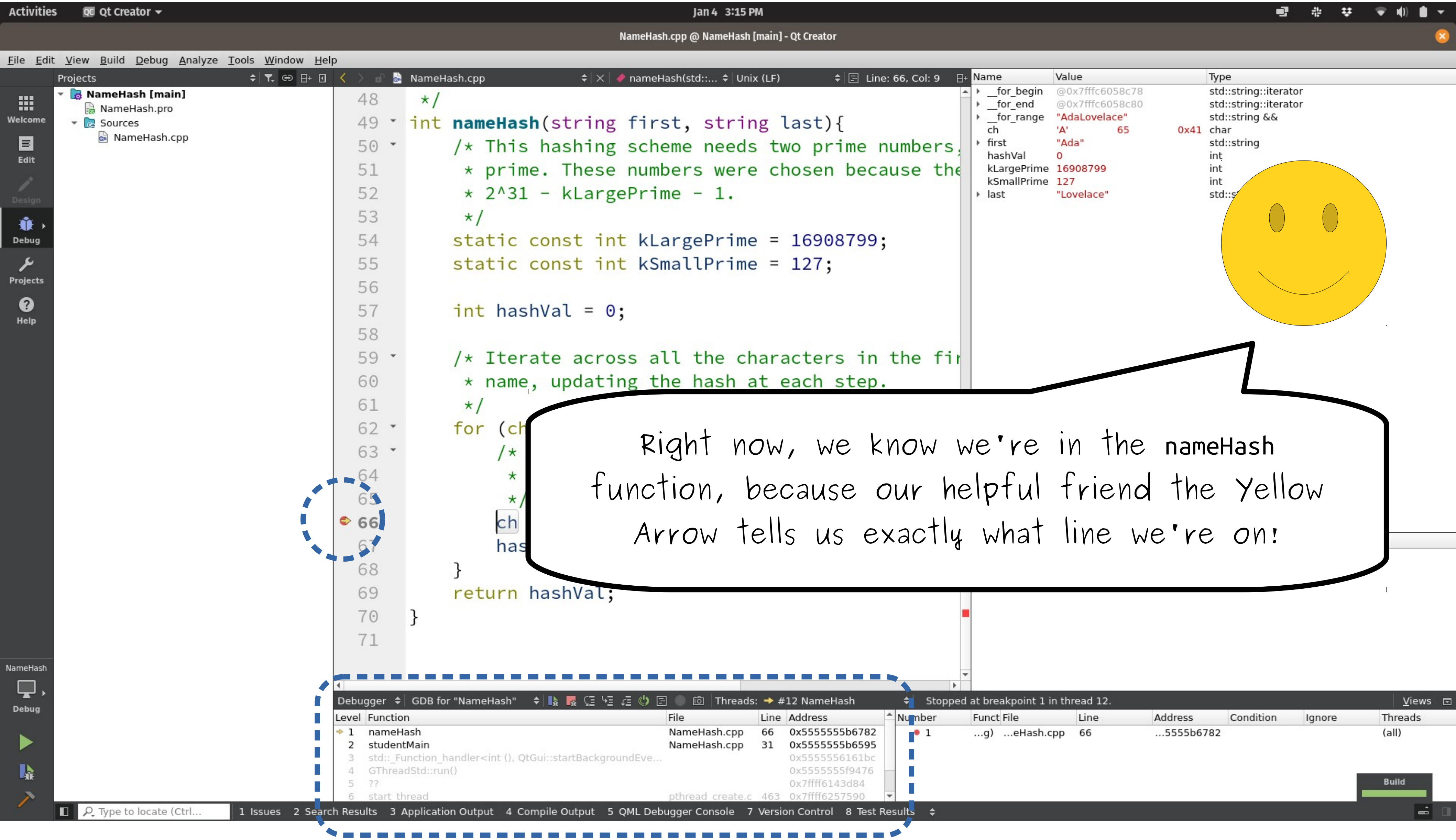
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std:::

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

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

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



```
    */
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 fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

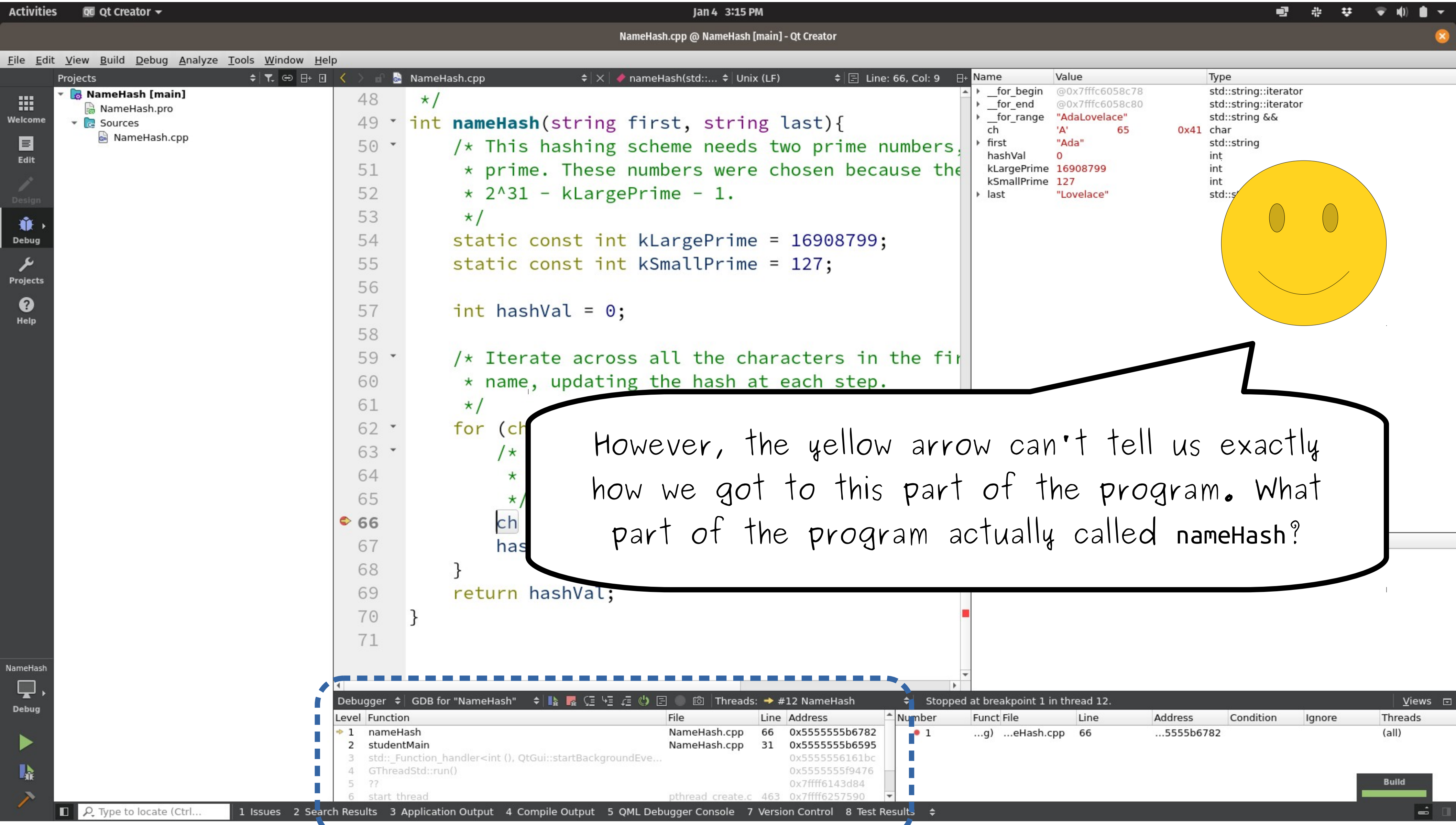
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::s

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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!



```
  */
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 fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

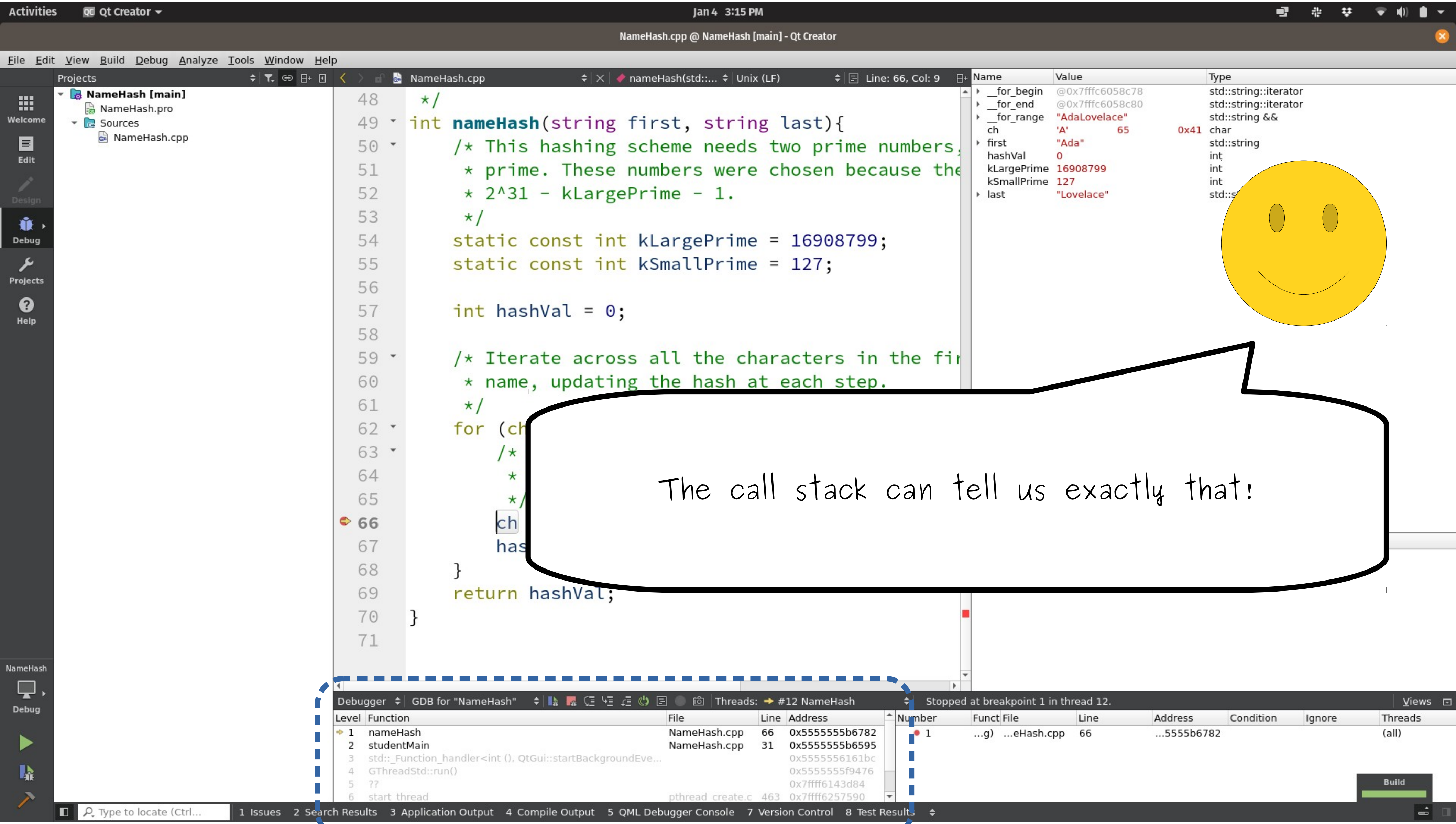
Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65 0x41	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::s

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

Stopped at breakpoint 1 in thread 12.

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

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**?



```
 */
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 fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

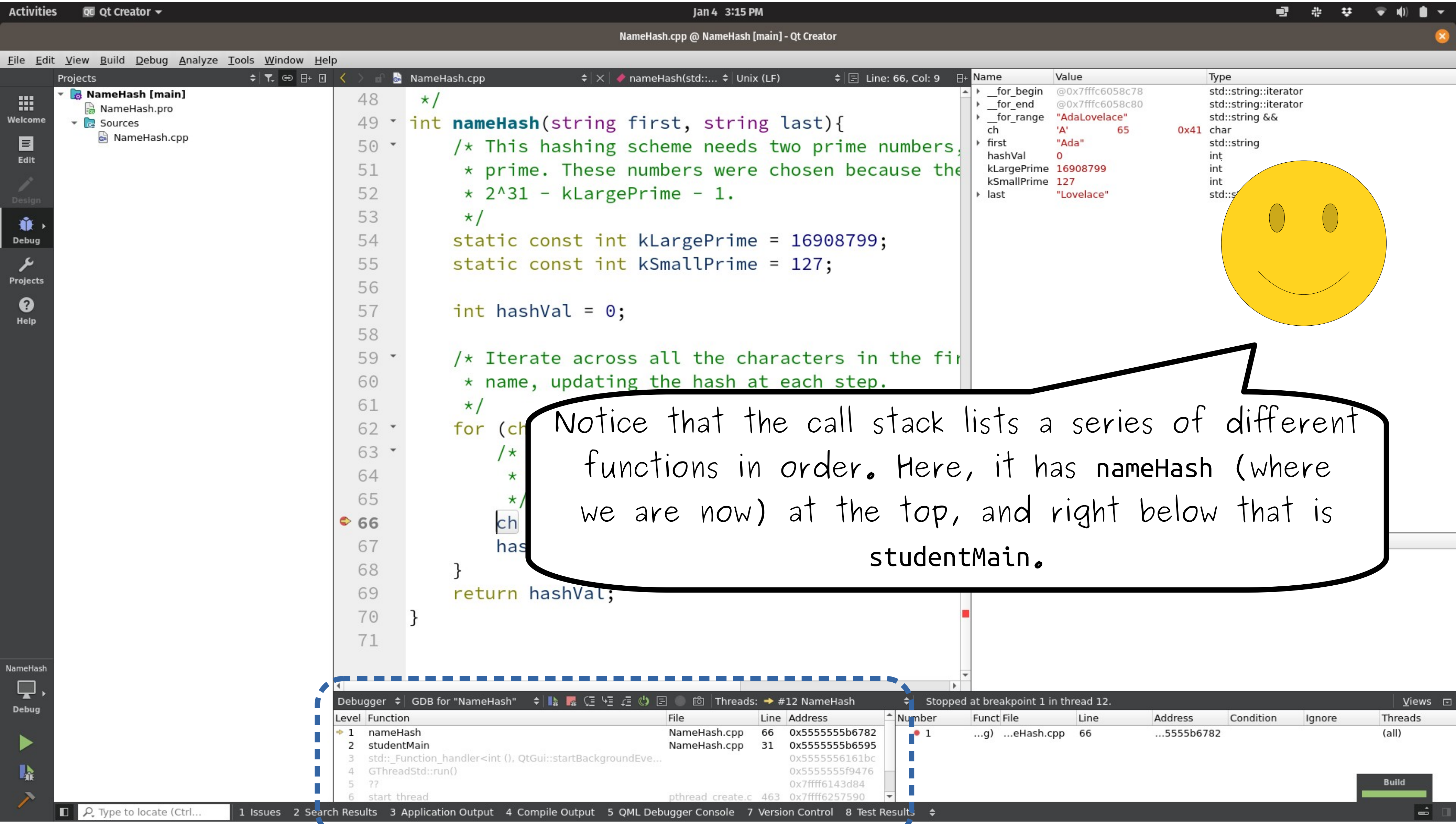
Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65 0x41	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::s

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

The call stack can tell us exactly that!



```
 */
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 fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

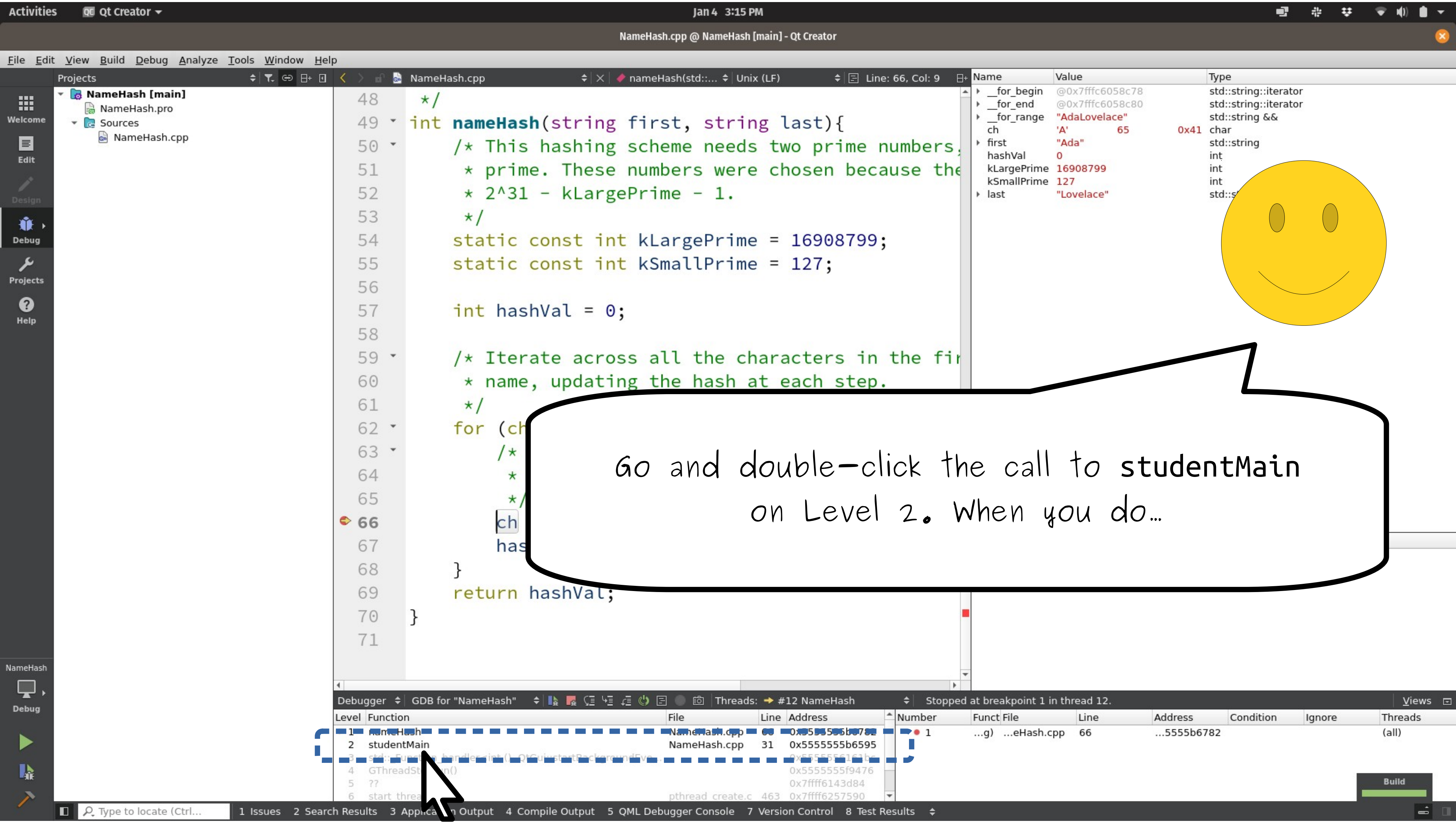
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::s

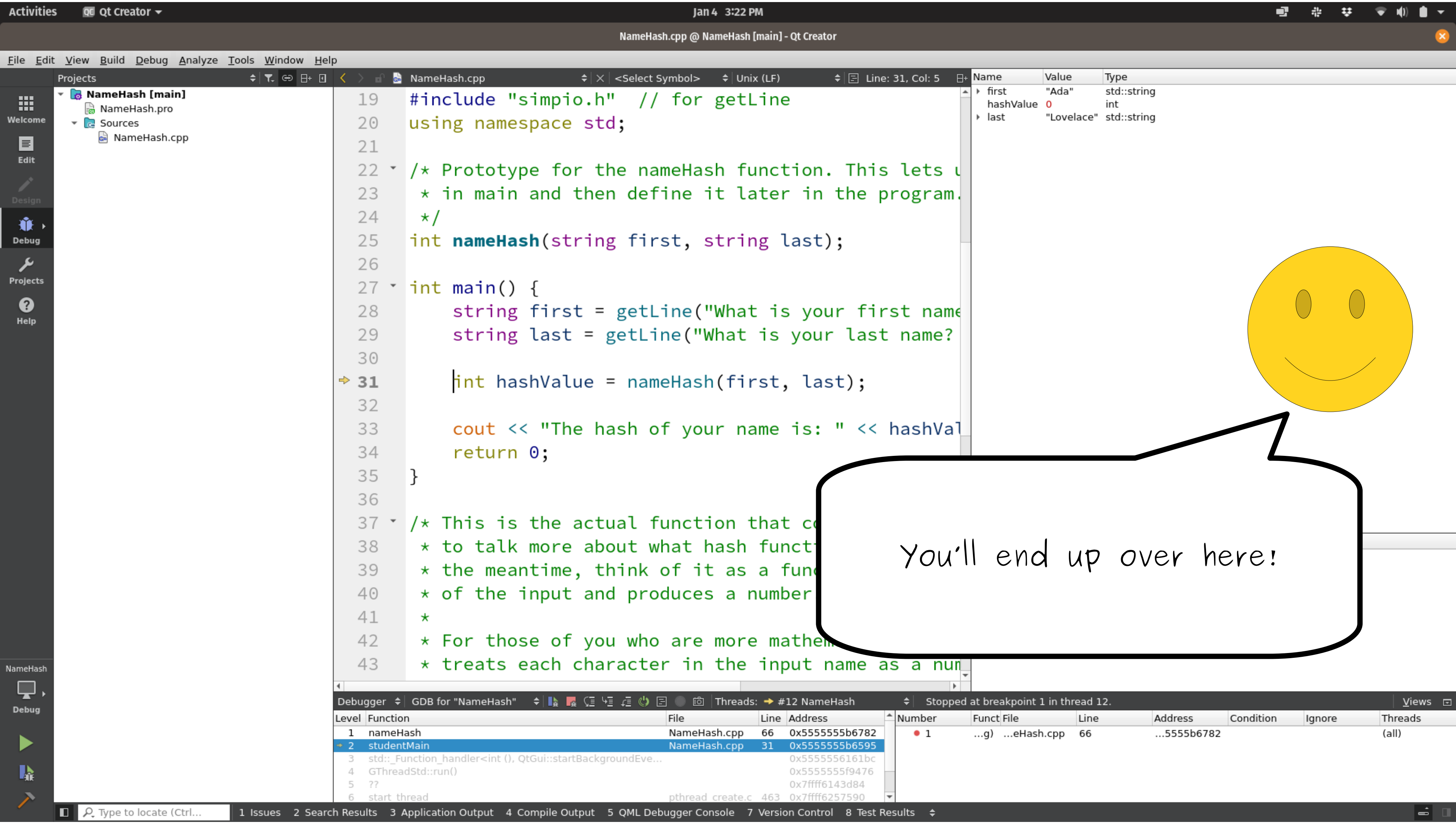
Stopped at breakpoint 1 in thread 12.

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

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

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**.





```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is the actual function that co
 * to talk more about what hash funct
 * the meantime, think of it as a func
 * of the input and produces a number
 *
 * For those of you who are more mathem
 * treats each character in the input name as a nun
```

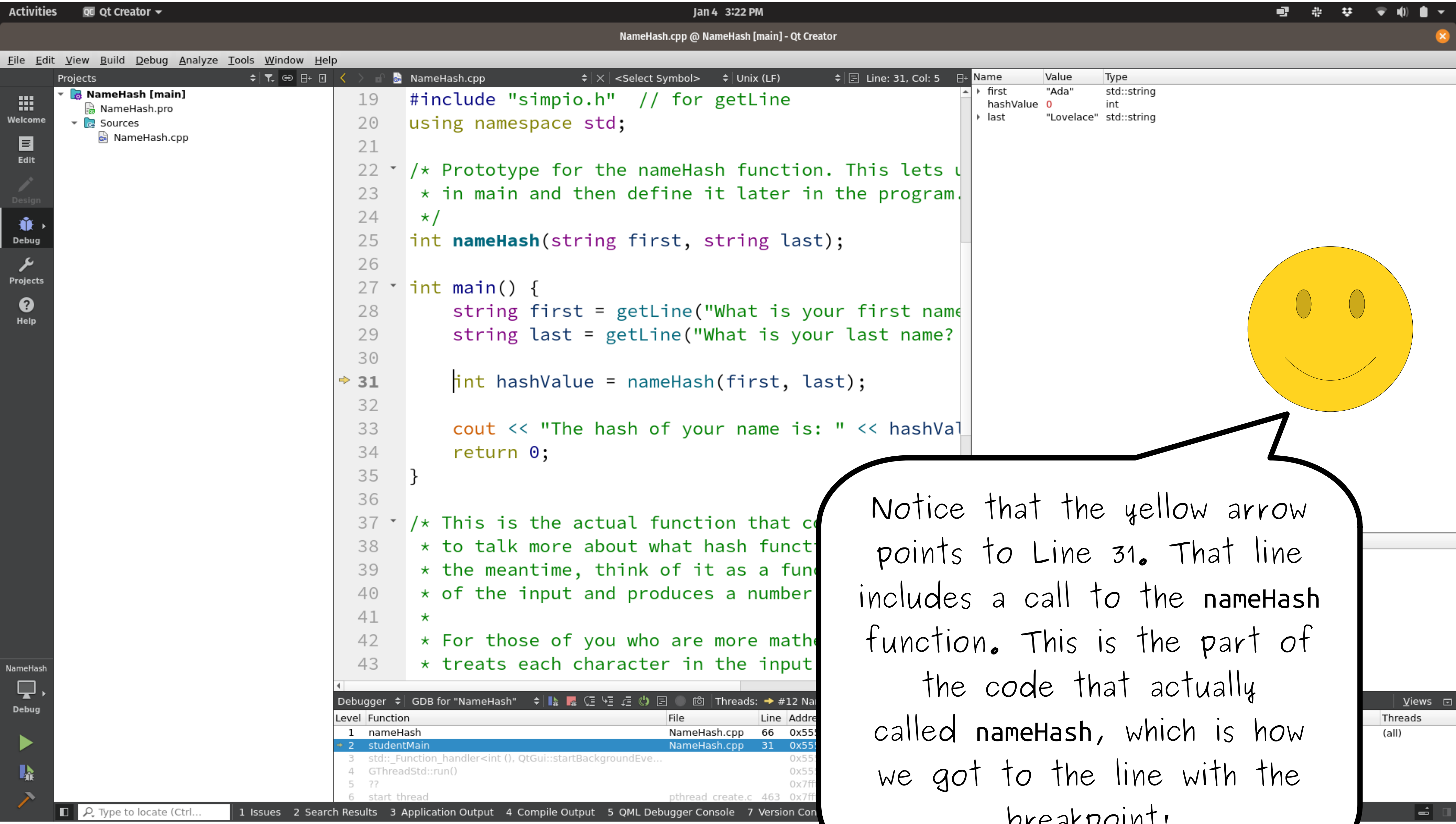
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

You'll end up over here!

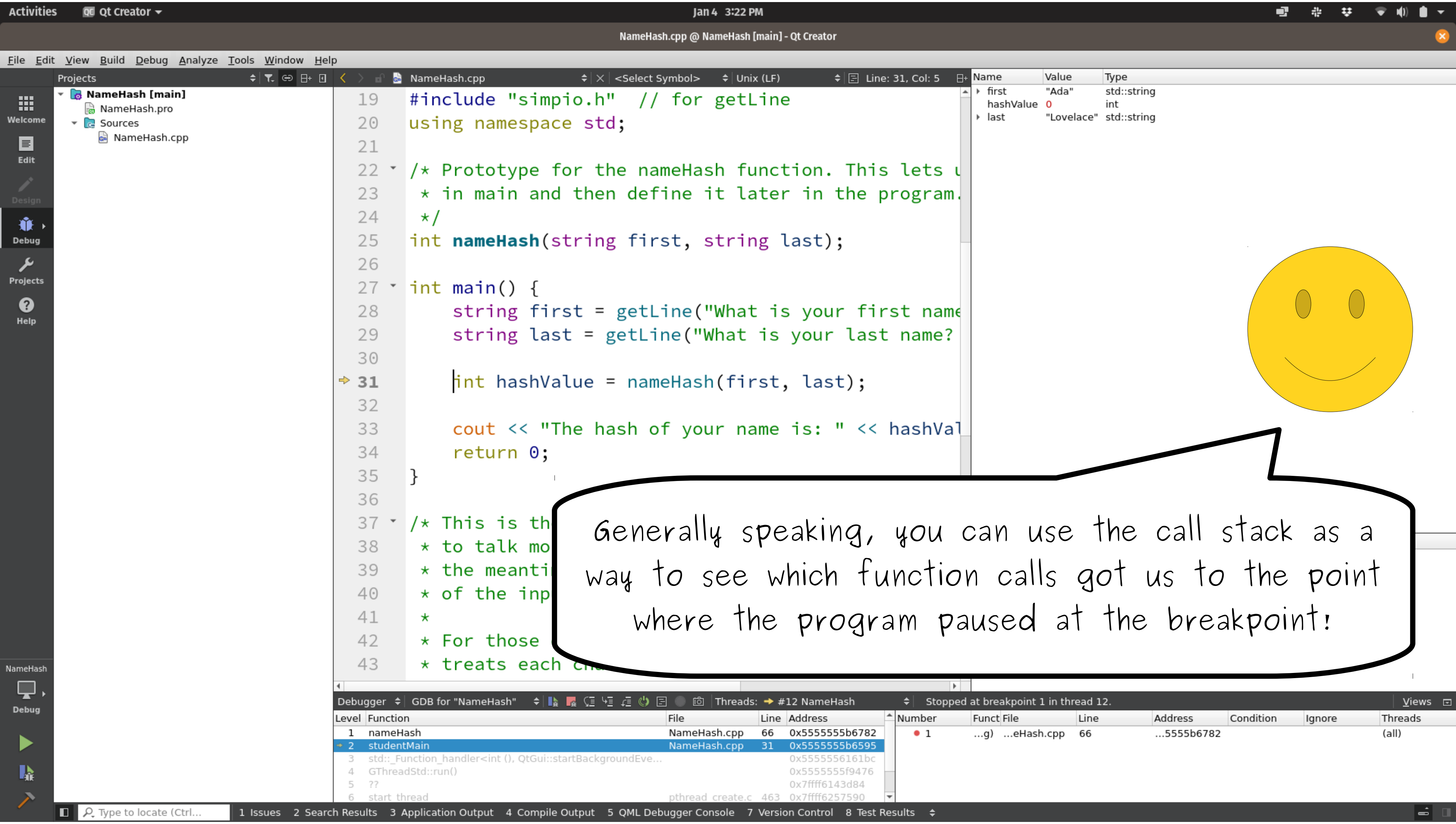


```
19  #include "simpio.h"  // for getLine
20  using namespace std;
21
22  /* Prototype for the nameHash function. This lets u
23   * in main and then define it later in the program.
24   */
25  int nameHash(string first, string last);
26
27  int main() {
28      string first = getLine("What is your 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: " << hashVal
34      return 0;
35  }
36
37  /* This is the actual function that c
38   * to talk more about what hash funct
39   * the meantime, think of it as a fun
40   * of the input and produces a number
41   *
42   * For those of you who are more mathe
43   * treats each character in the input
```

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

Level	Function	File	Line	Addre
1	nameHash	NameHash.cpp	66	0x55
2	studentMain	NameHash.cpp	31	0x55
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x55
4	GThreadStd::run()			0x55
5	??			0x7ff
6	start_thread	pthread_create.c	463	0x7ff

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!



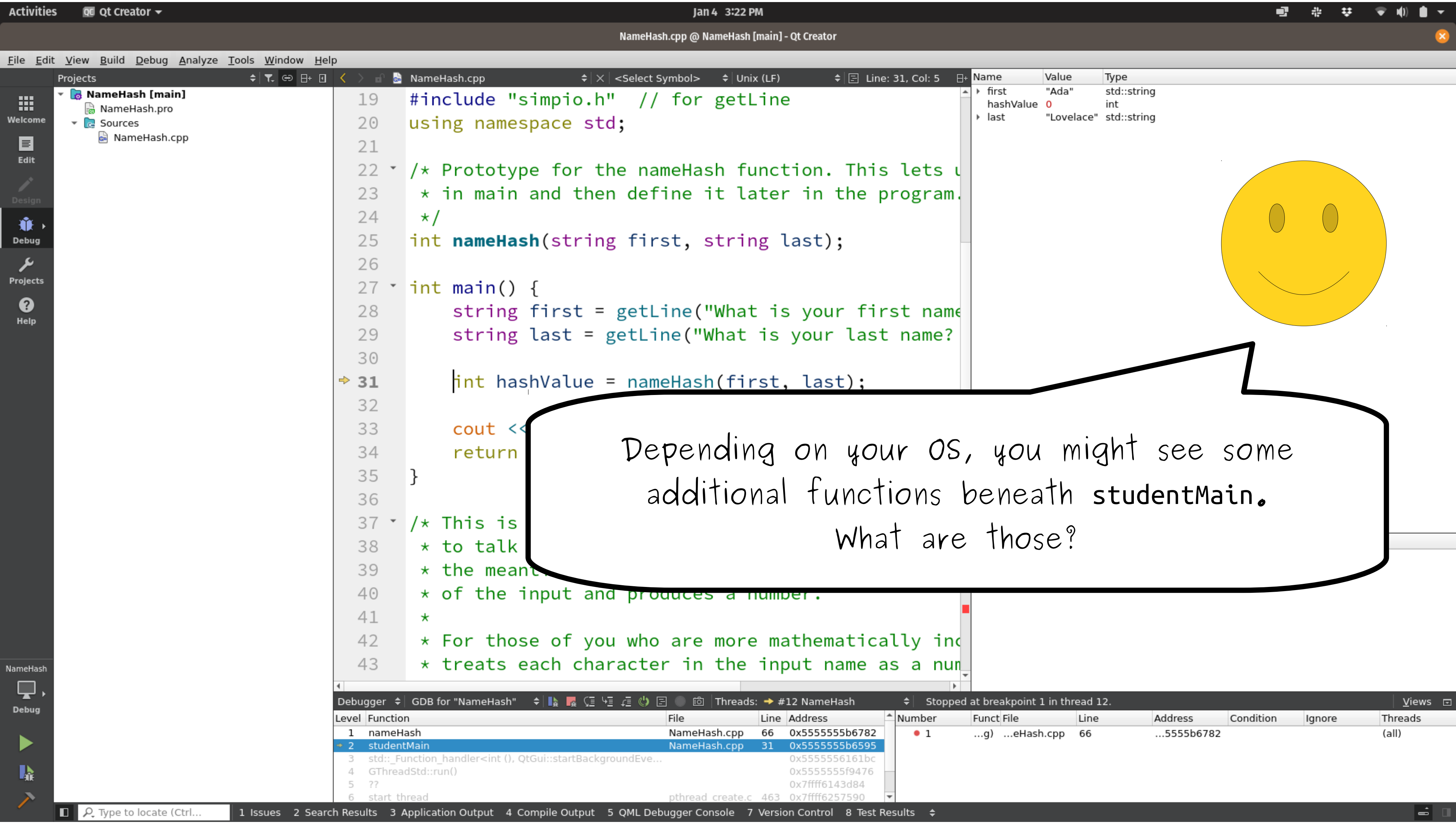
```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is th
 * to talk mo
 * the meanti
 * of the inp
 *
 * For those
 * treats each ch
```

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

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

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

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!



```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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 <<
    return
}

/* This is
 * to talk
 * the meani
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a num
```

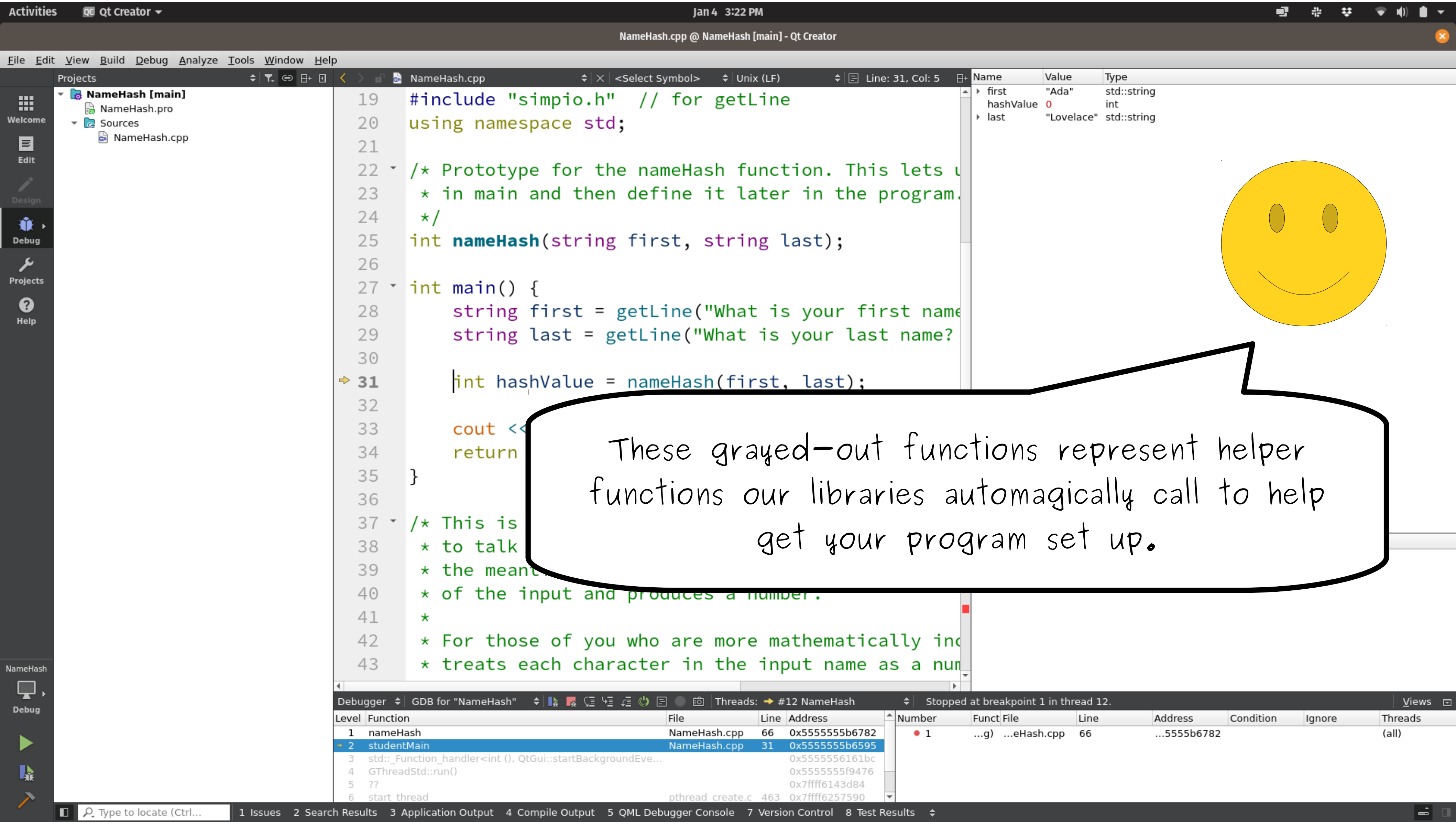
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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



```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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 <<
    return
}

/* This is
 * to talk
 * the meani
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a nun
```

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

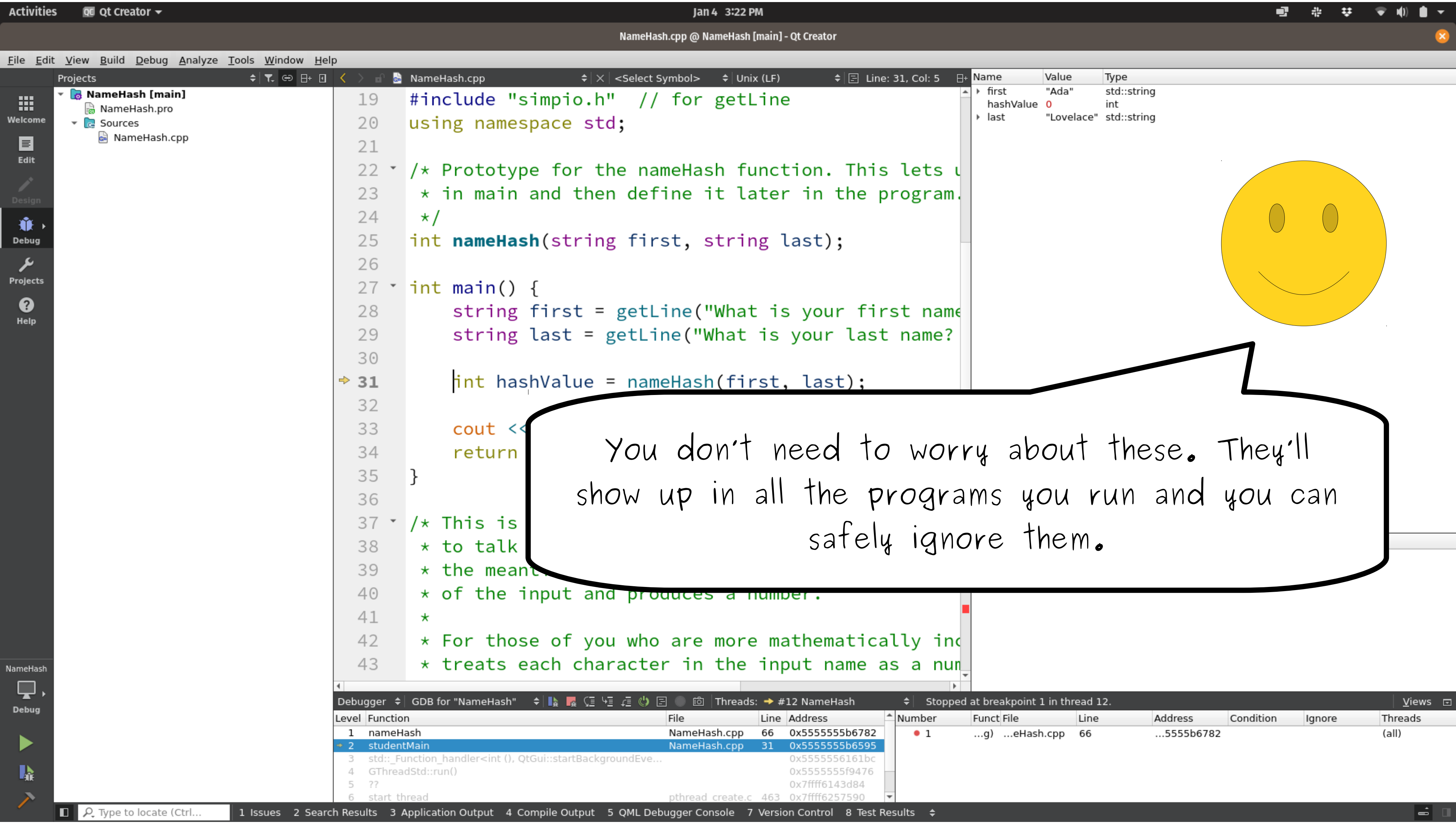
Debugger   GDB for "NameHash"   Threads: #12 NameHash   Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x555555616lbc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

Views
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

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



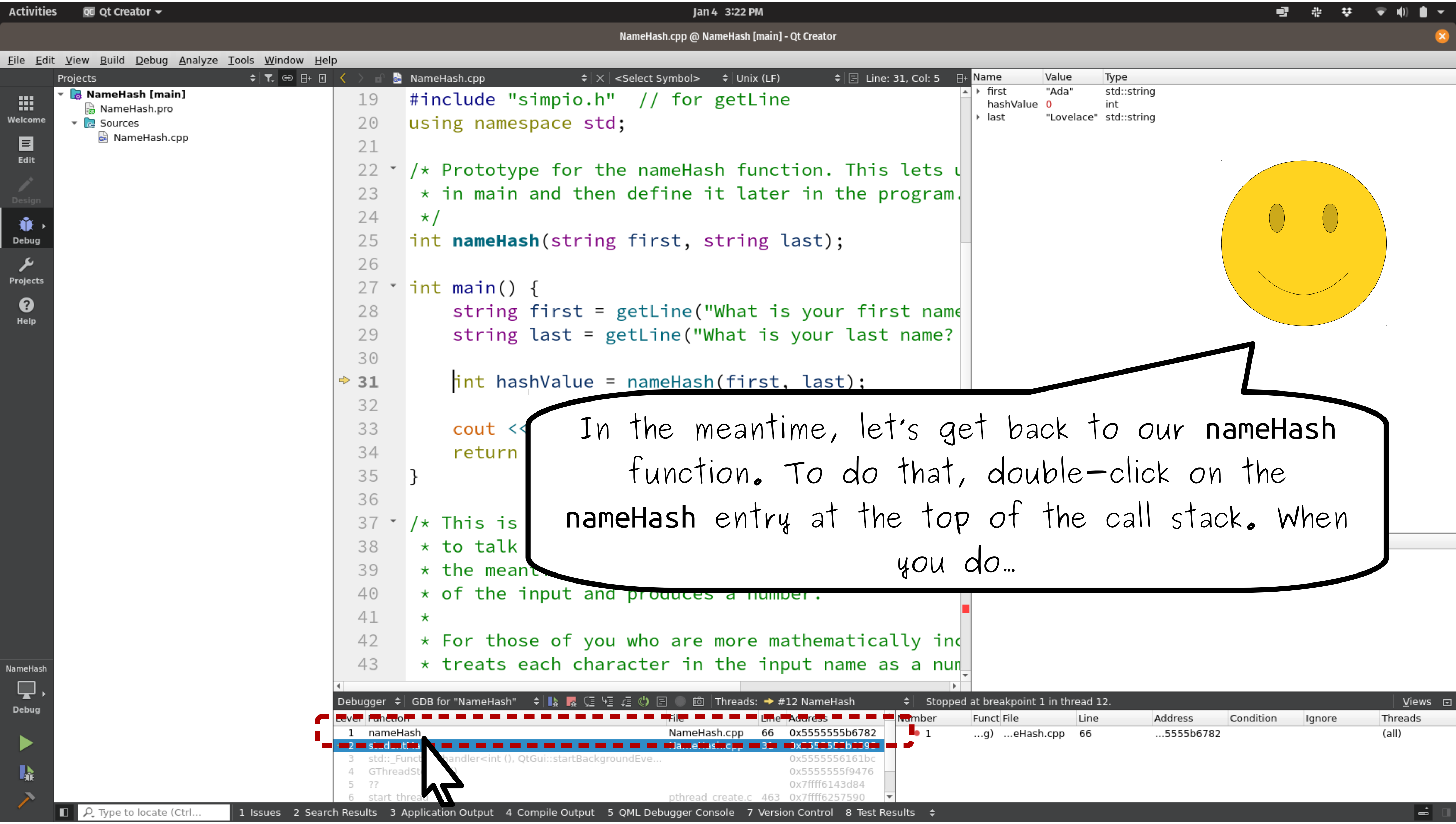
```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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 <<
    return
}

/* This is
 * to talk
 * the mean
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a num
```

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

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

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

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



```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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 <<
    return
}

/* This is
 * to talk
 * the meant
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a num
```

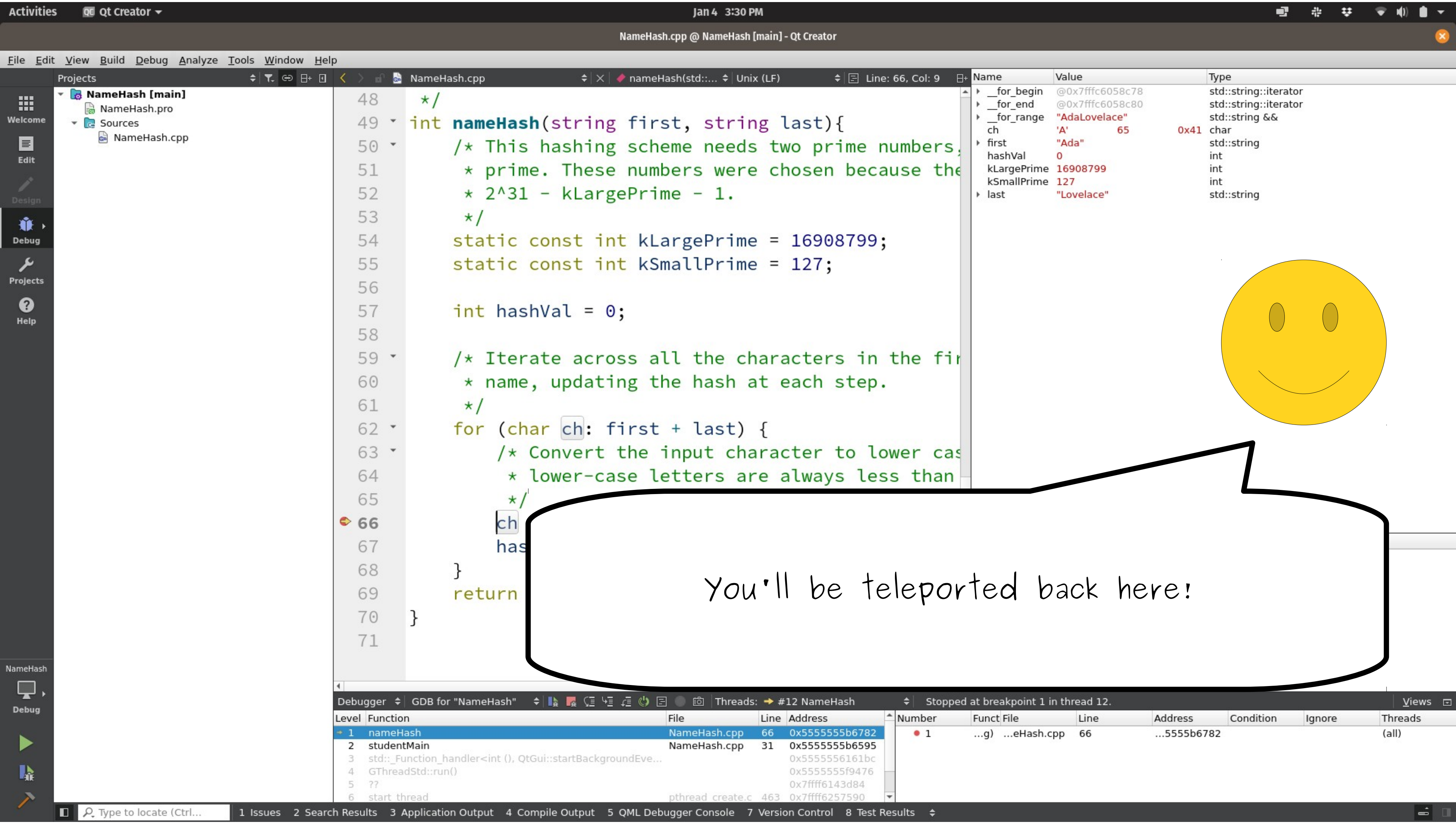
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

Stopped at breakpoint 1 in thread 12.

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	[illegible]	NameHash.cpp	31	[illegible]
3	std::_Funct...handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadSt...			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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…



```
  */
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 fir
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch
        has
    }
    return
}
```

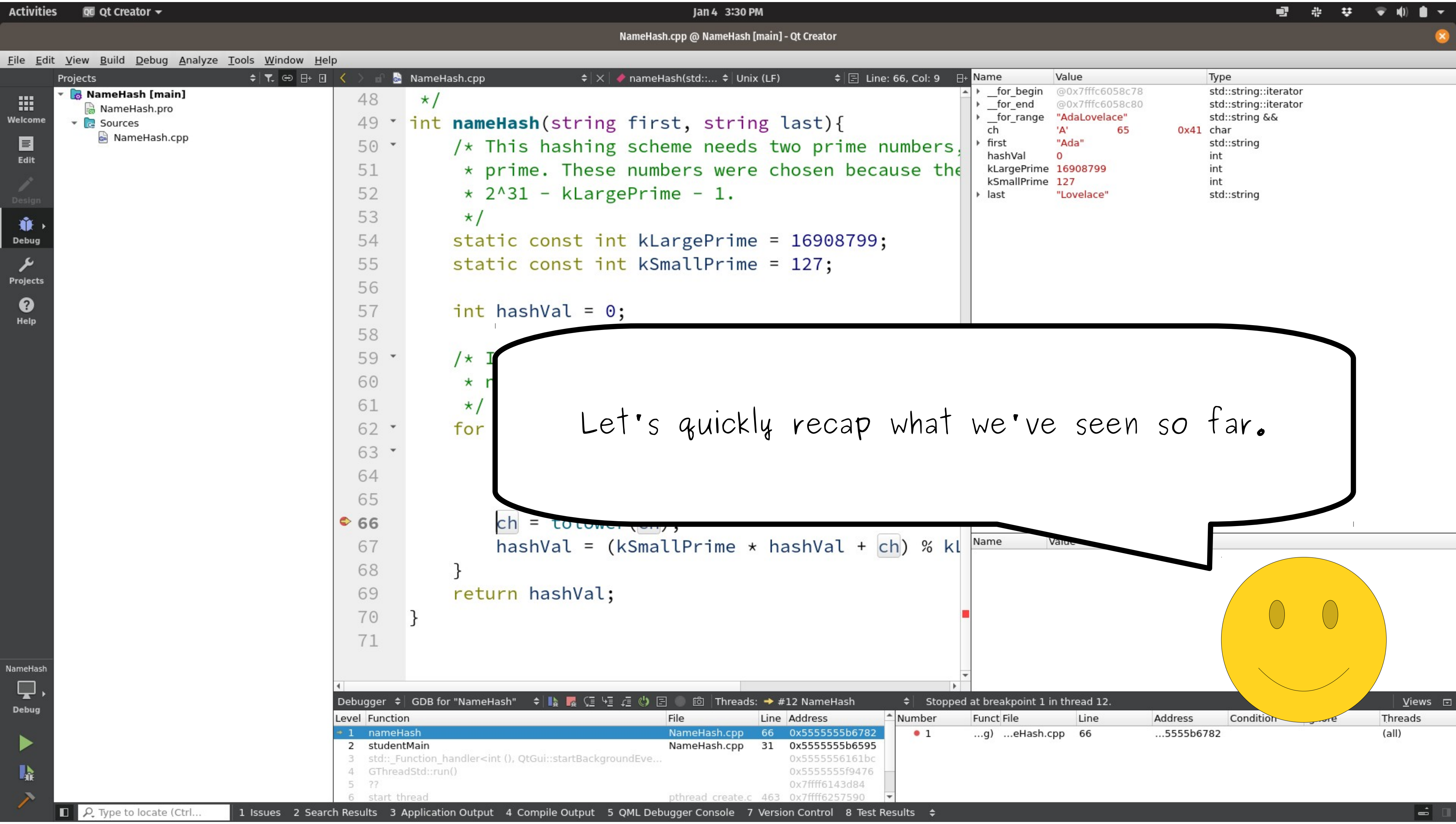
Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65 0x41	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

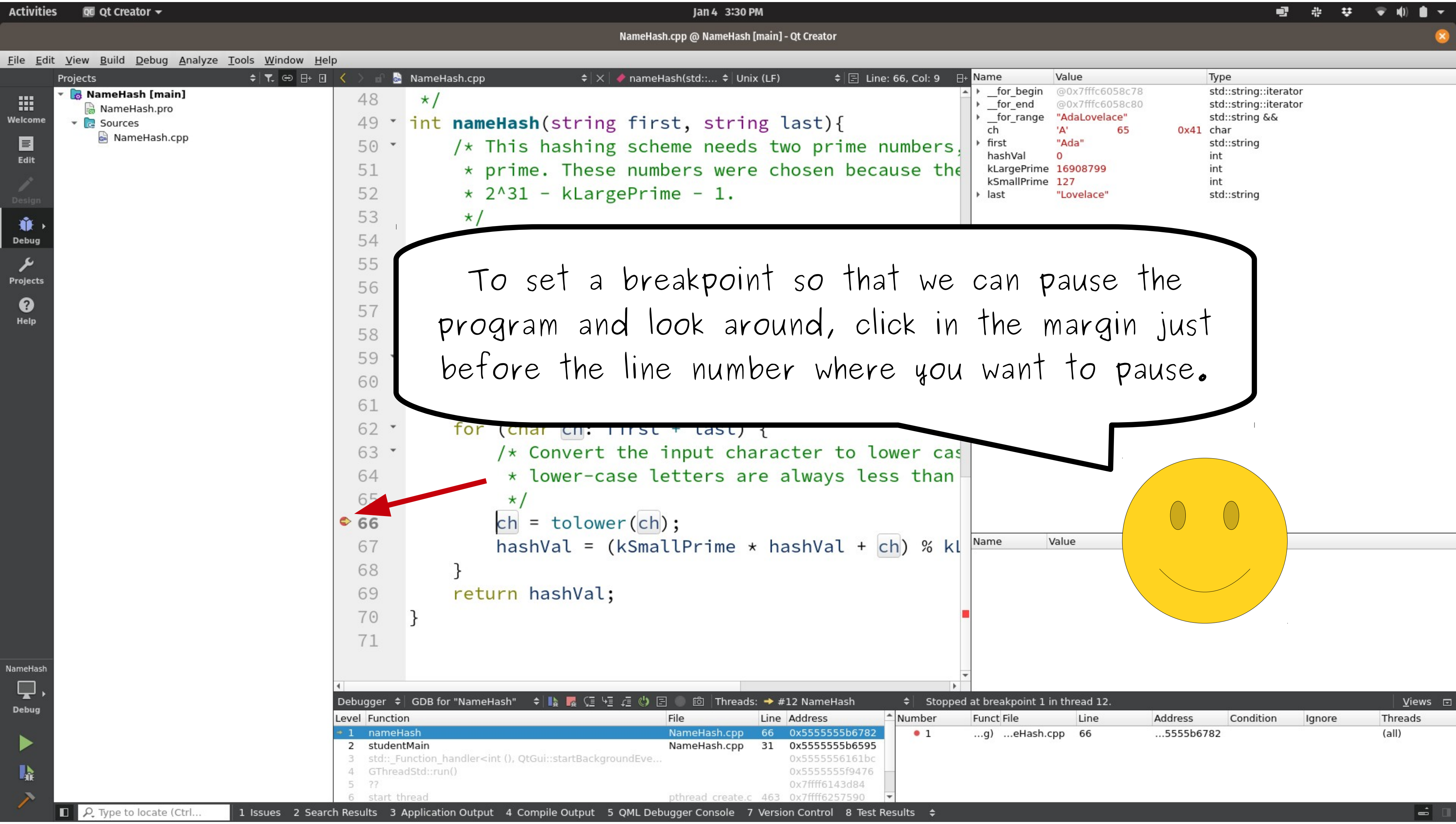
Stopped at breakpoint 1 in thread 12.

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

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

You'll be teleported back here!





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

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

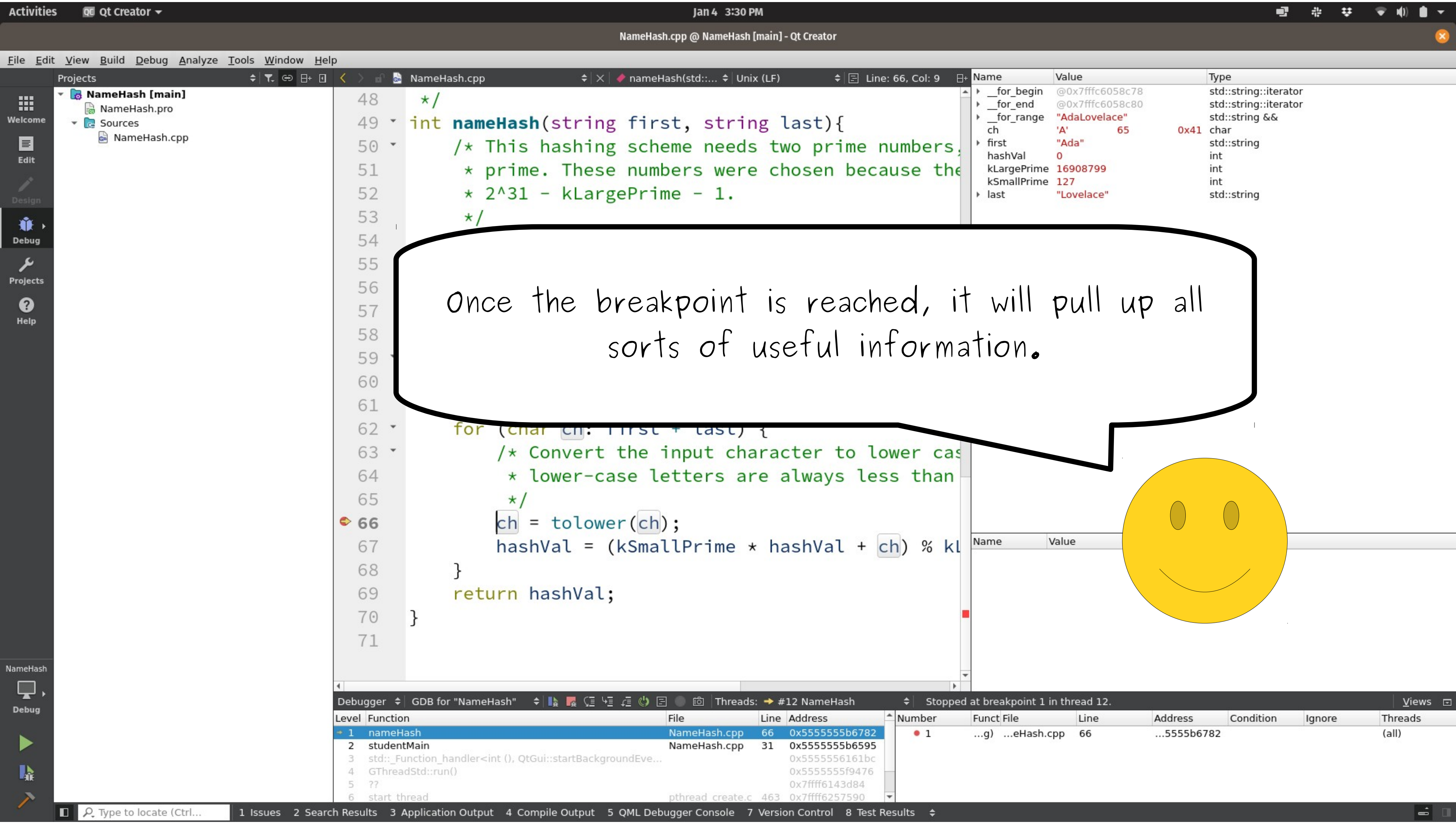
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Stopped at breakpoint 1 in thread 12.

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

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

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.



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

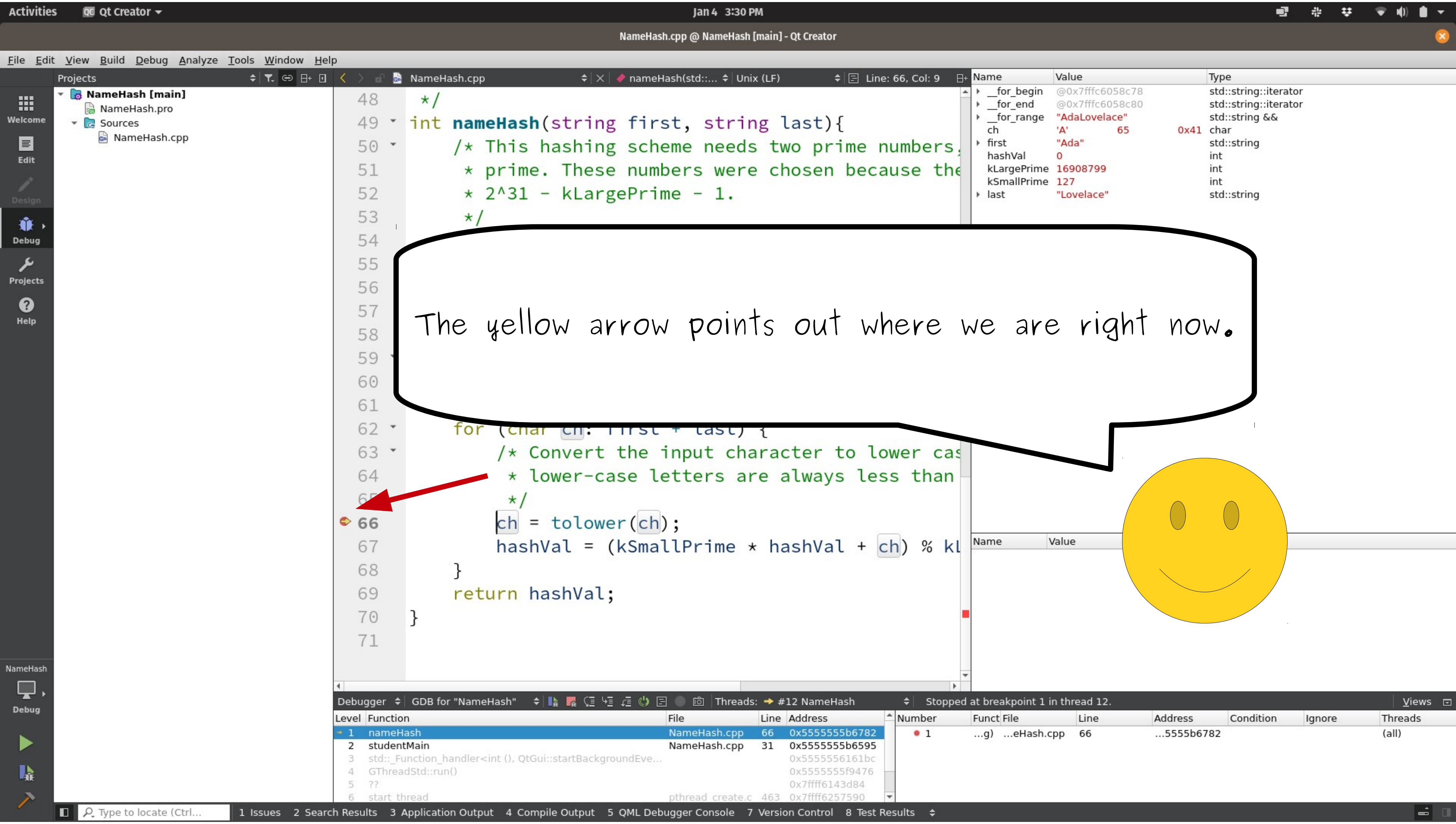
```
     */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */

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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	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
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

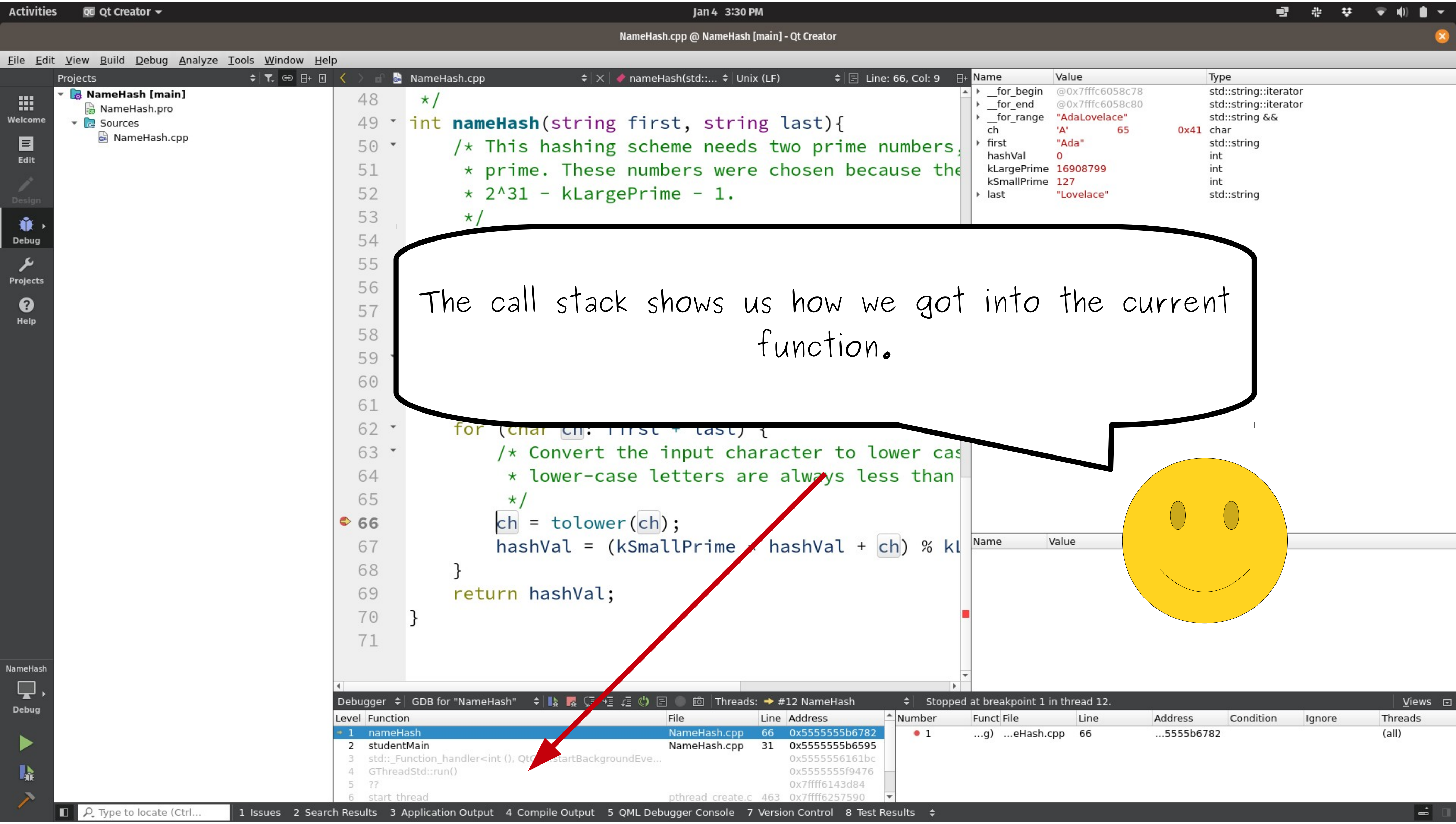


```
  */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */

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

```
Name          Value                    Type
__for_begin   @0x7fffc6058c78          std::string::iterator
__for_end     @0x7fffc6058c80          std::string::iterator
__for_range   "AdaLovelace"            std::string &&
ch            'A'      65      0x41    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 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::startBackgroundEve...                    0x5555556161bc
4     GThreadStd::run()                                                             0x5555555f9476
5     ??                                                                            0x7ffff6143d84
6     start_thread                                            pthread_create.c  463  0x7ffff6257590

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

The yellow arrow points out where we are right now.



NameHash.cpp @ NameHash [main] - Qt Creator

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

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

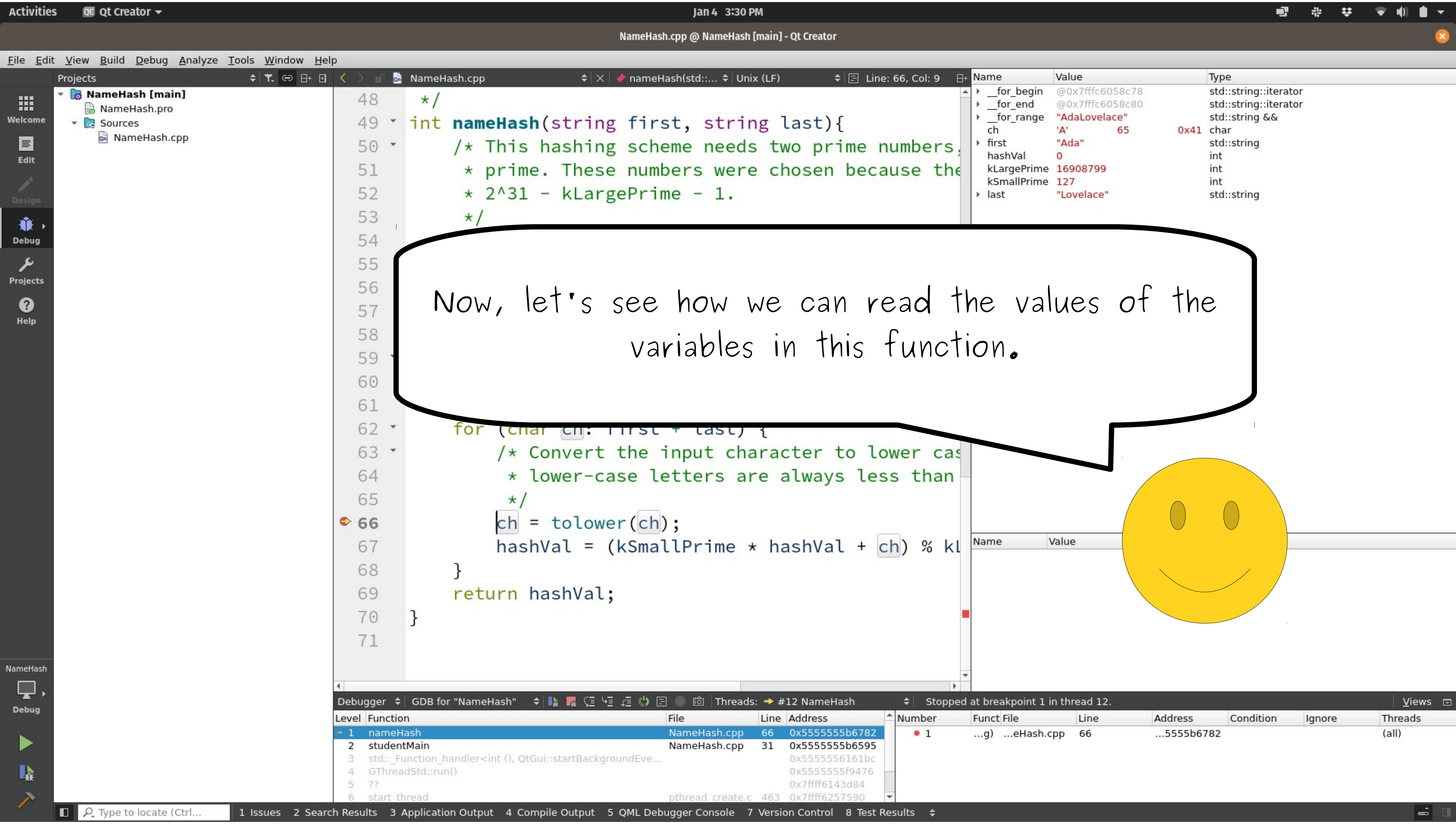
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped 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 (), QtC...startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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



```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    for (char ch : first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

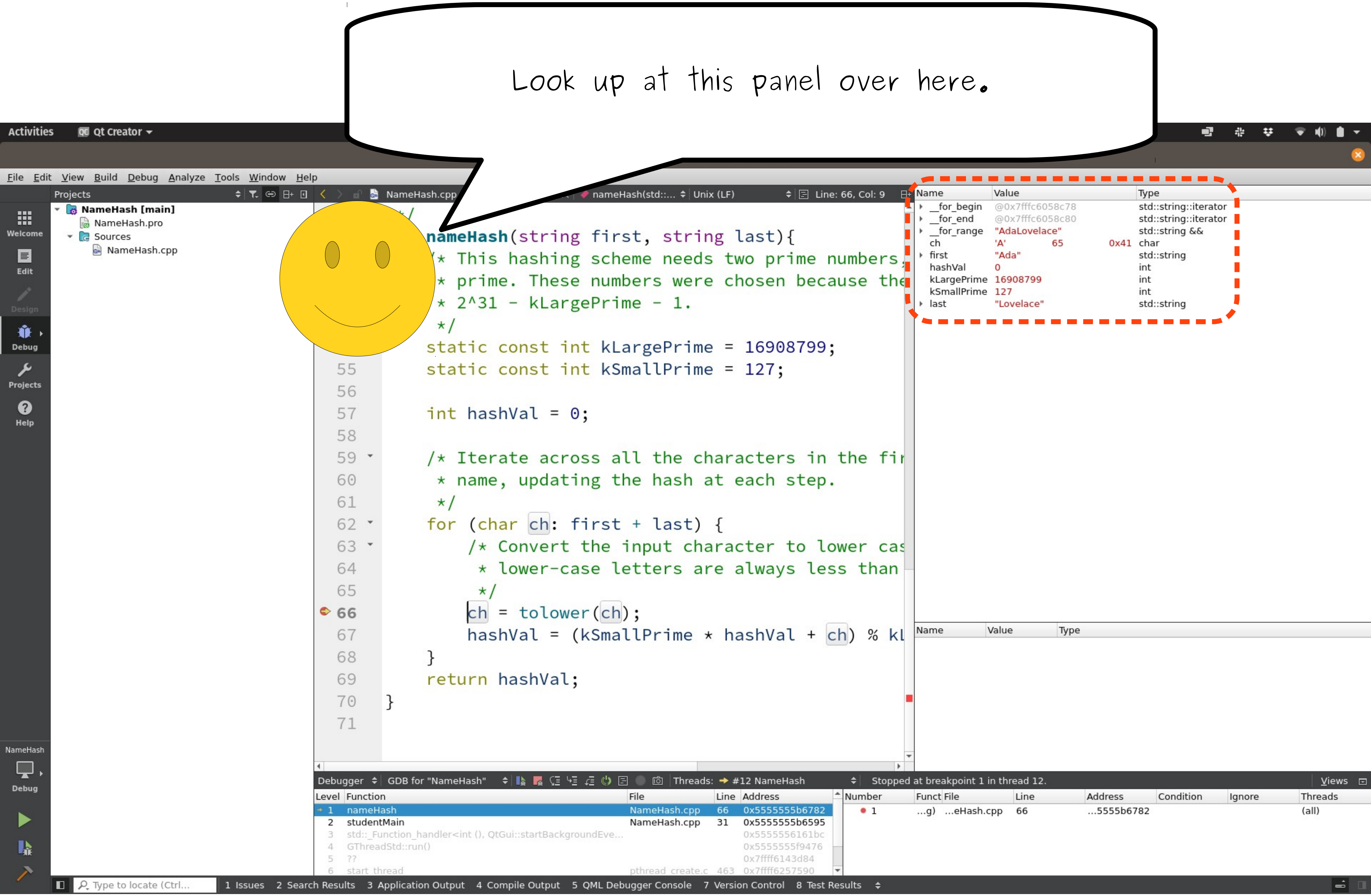
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Stopped at breakpoint 1 in thread 12.

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

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

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



Look up at this panel over here.

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

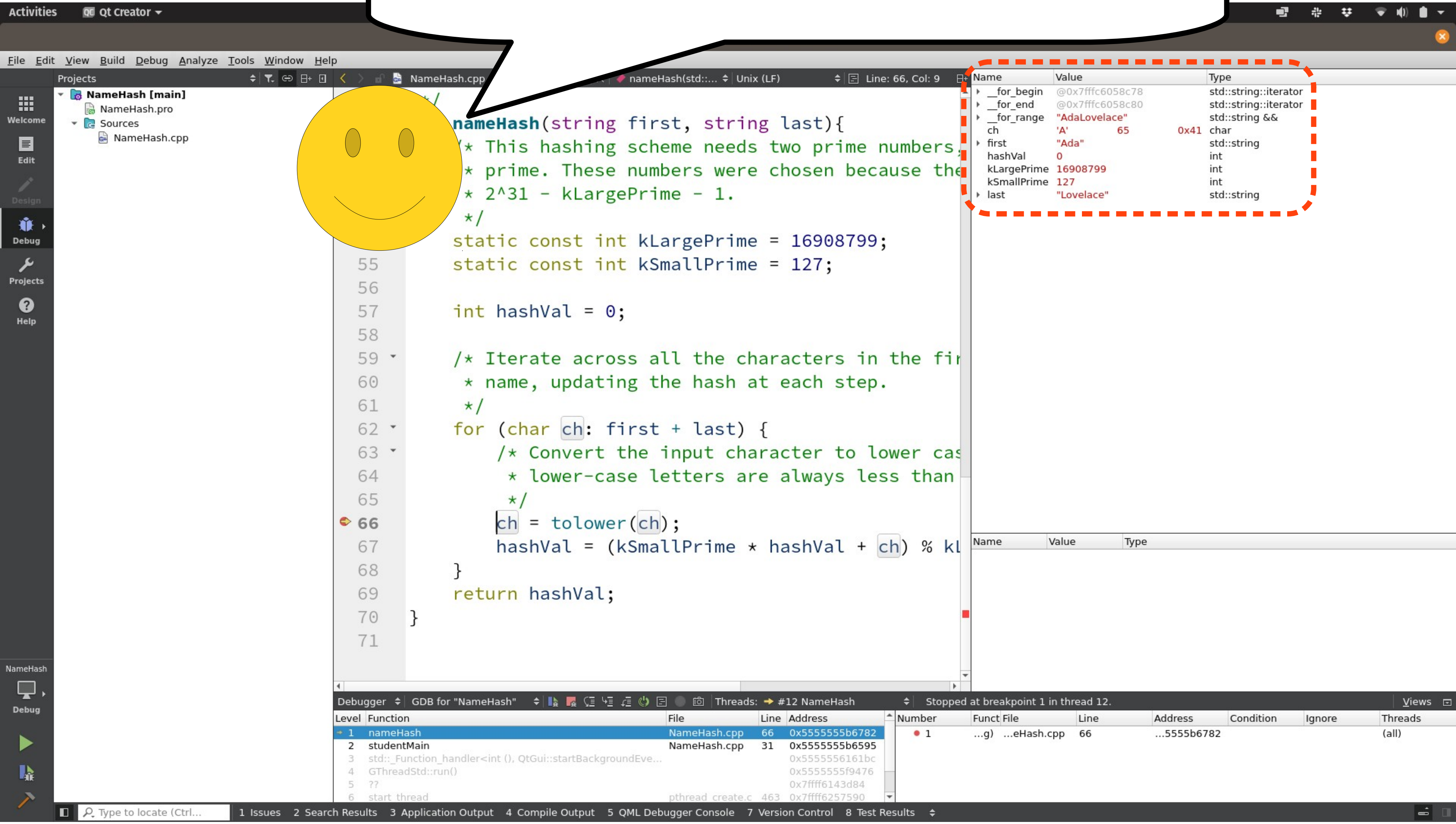
Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65 0x41	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Stopped at breakpoint 1 in thread 12.

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

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

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



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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

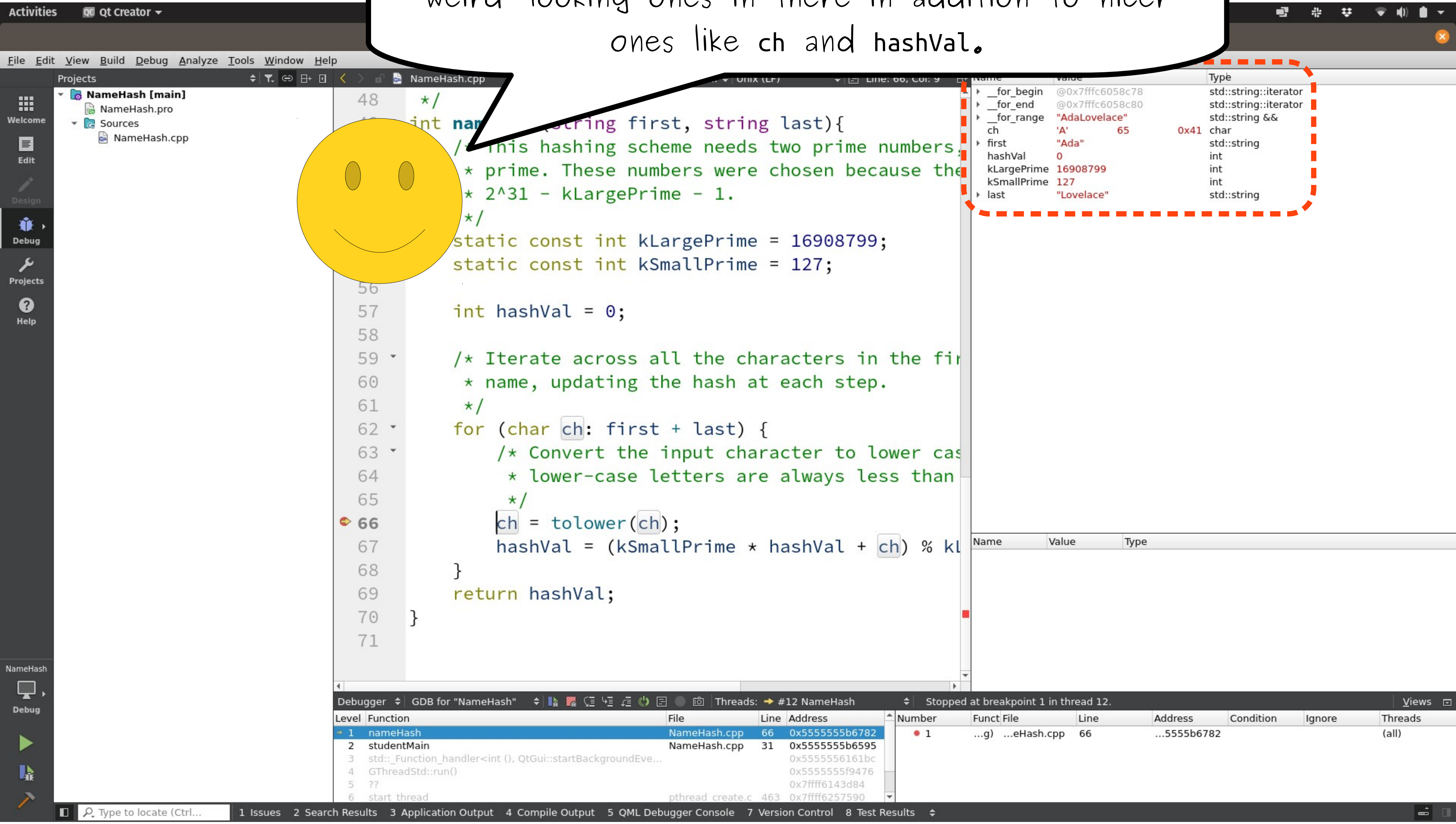
Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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**.



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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

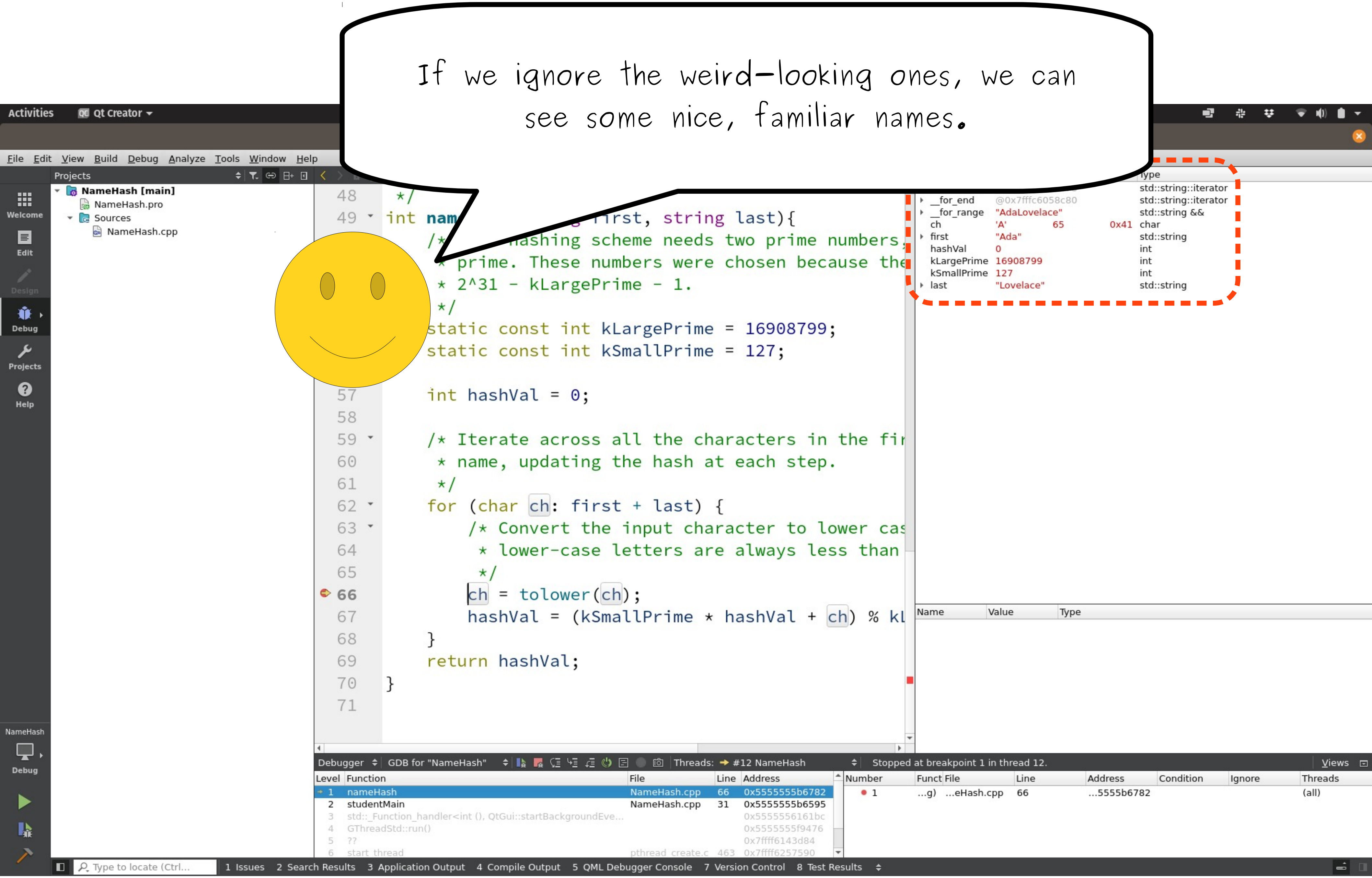
Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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


Activities
Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Debugger
GDB for "NameHash"
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 7 Version Control 8 Test Results

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

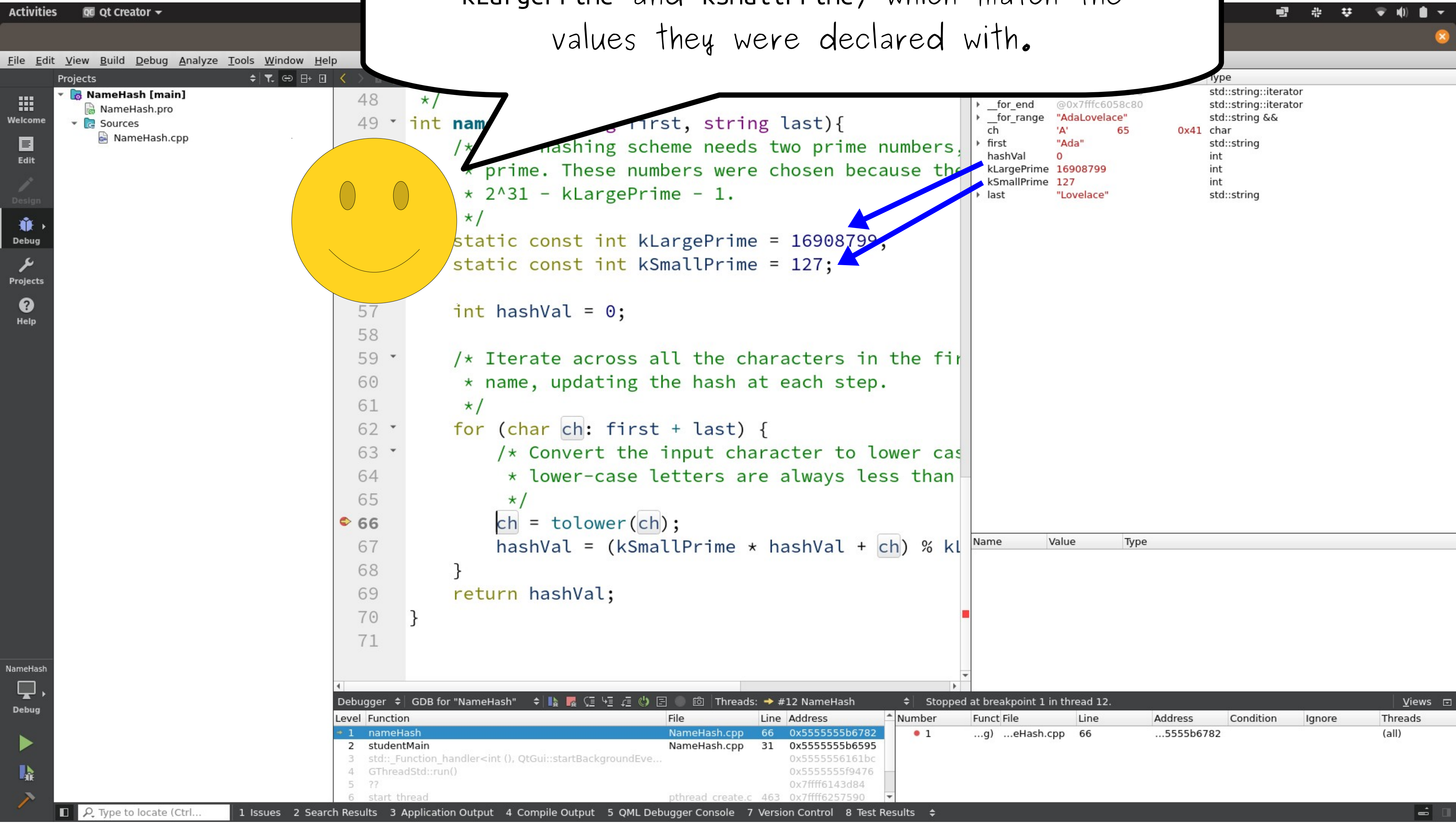
		ype
		std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65 0x41	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Name	Value	Type

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

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

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


Activities
Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
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

```
   */
int nam        irst, string last){
    /*     hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because th
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

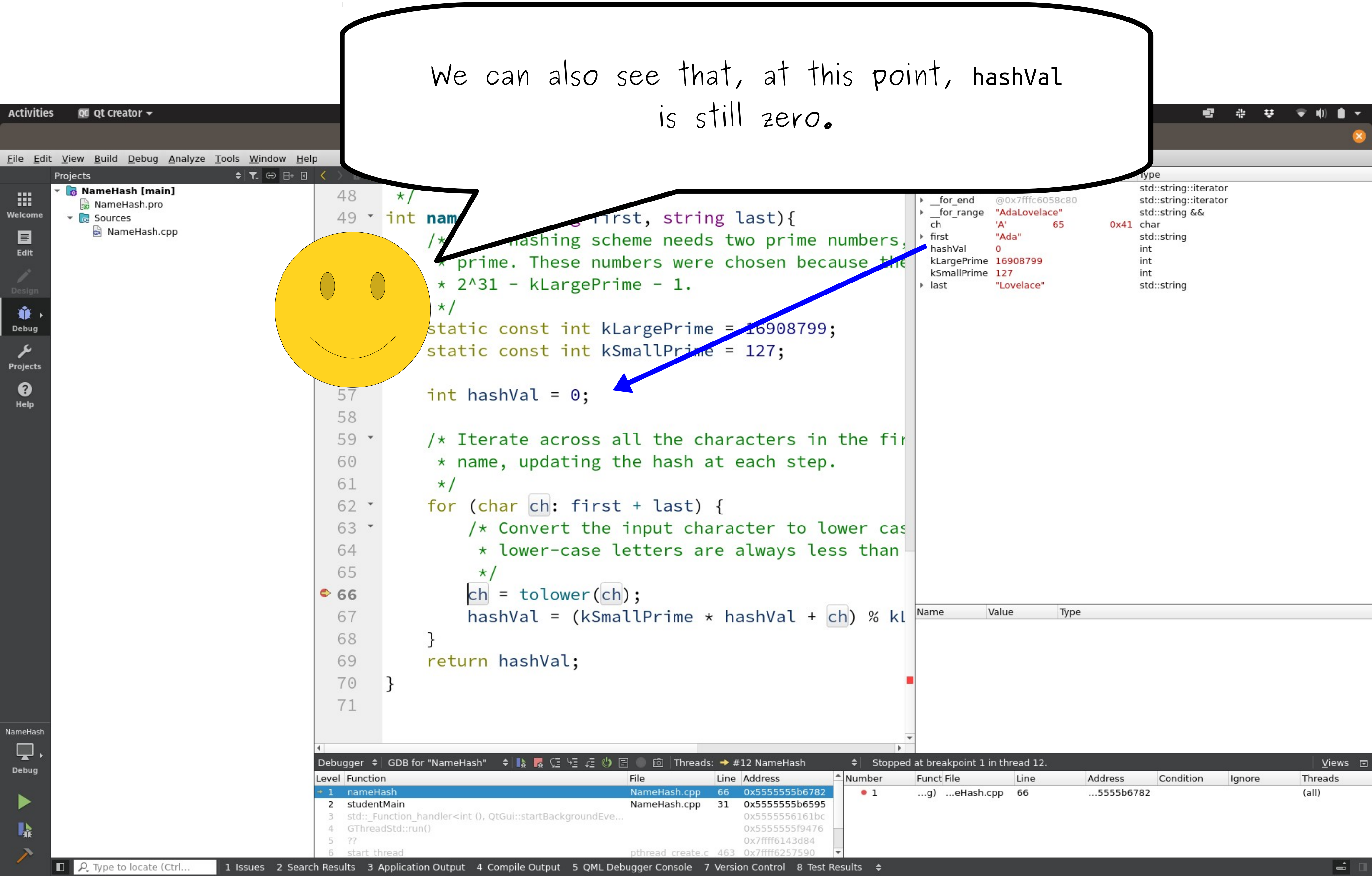
				Type
				std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



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

```
 */
int nam      irst, string last){
    /*    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 fir
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

Name	Value			Type
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string::iterator
ch	'A'	65	0x41	std::string &&
first	"Ada"			char
hashVal	0			std::string
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			int
				std::string

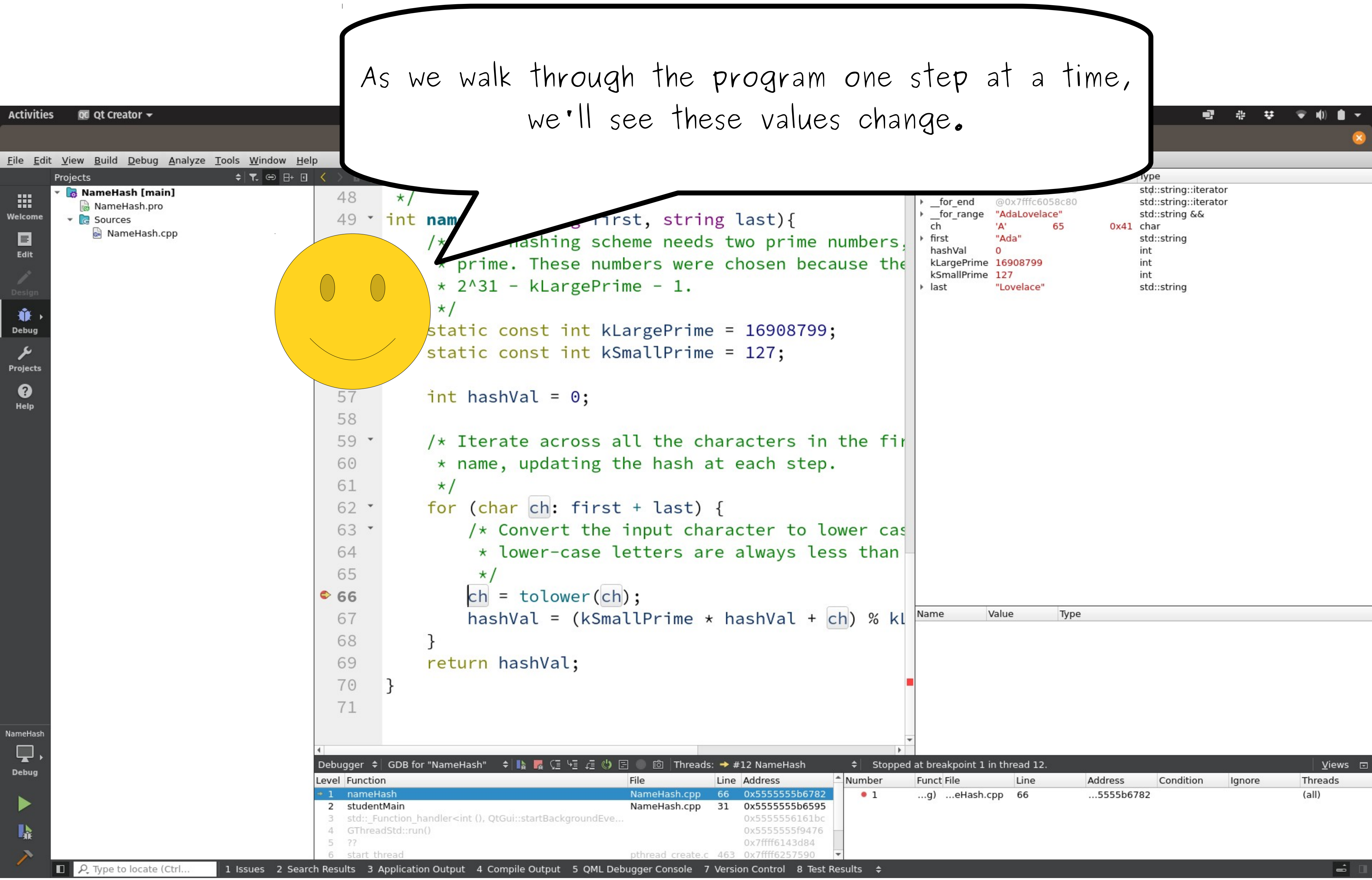
Name	Value	Type

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

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

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

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



```
 */
int nam         irst, string last){
    /*    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 fi
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

			Type
			std::string::iterator
__for_end	@0x7fffc6058c80		std::string::iterator
__for_range	"AdaLovelace"		std::string &&
ch	'A' 65	0x41	char
first	"Ada"		std::string
hashVal	0		int
kLargePrime	16908799		int
kSmallPrime	127		int
last	"Lovelace"		std::string

Name	Value	Type

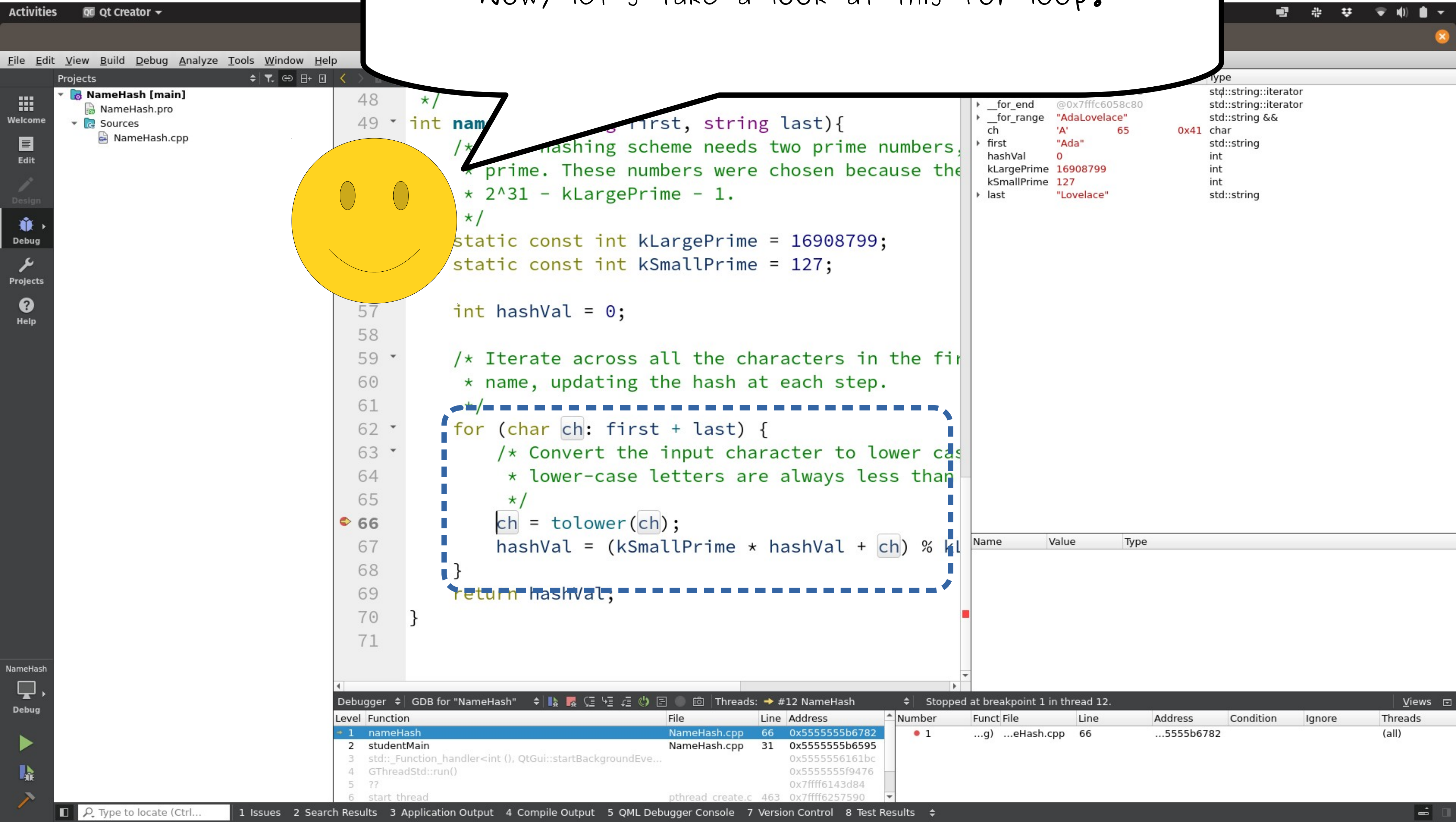
Debugger GDB for "NameHash" Threads: #12 NameHash Stopped at breakpoint 1 in thread 12. Views

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
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

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



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

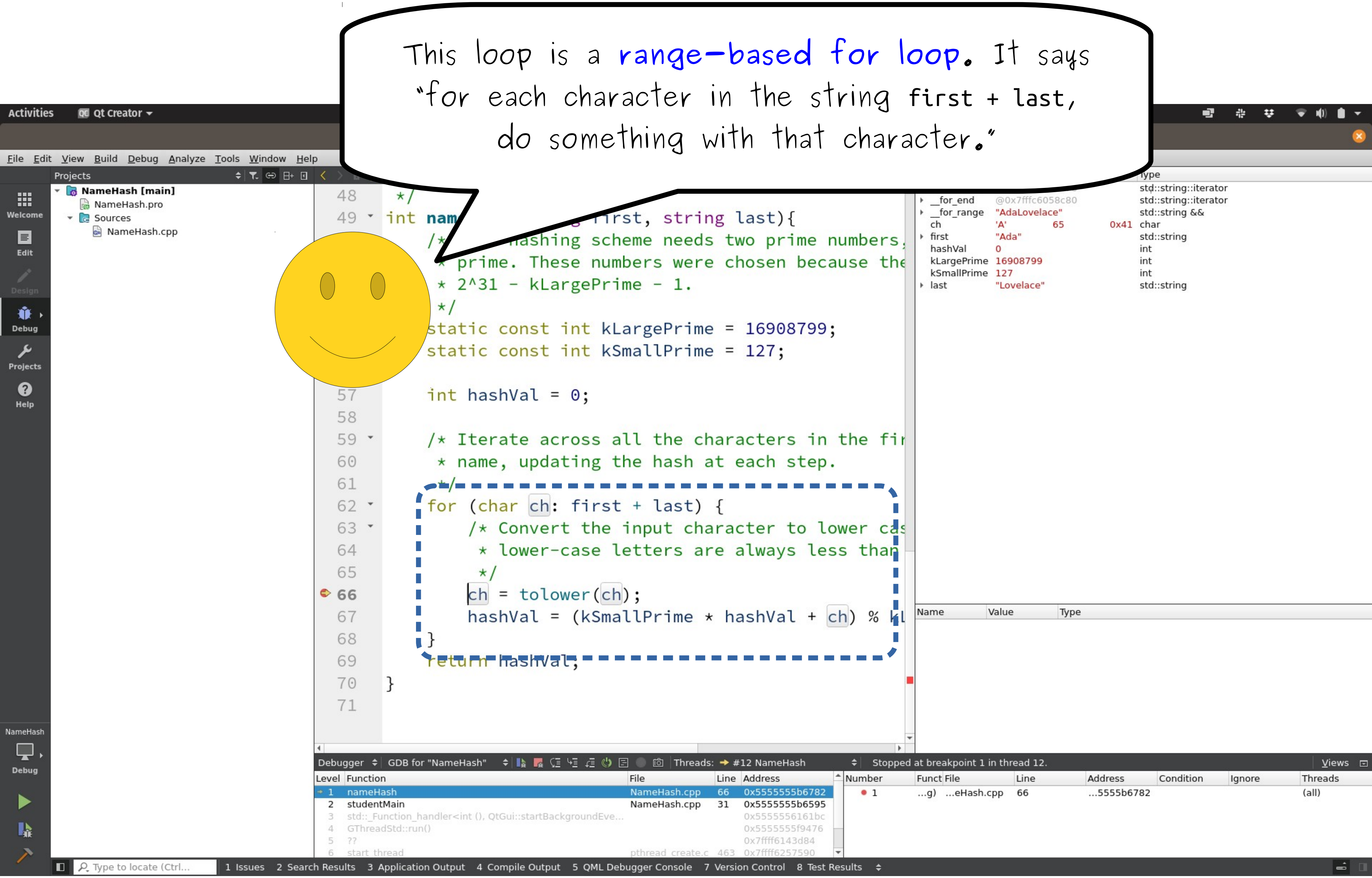
Name	Value			Type
				std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



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

```
 */
int nam           irst, string last){
    /*          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 fi
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

				Type
				std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

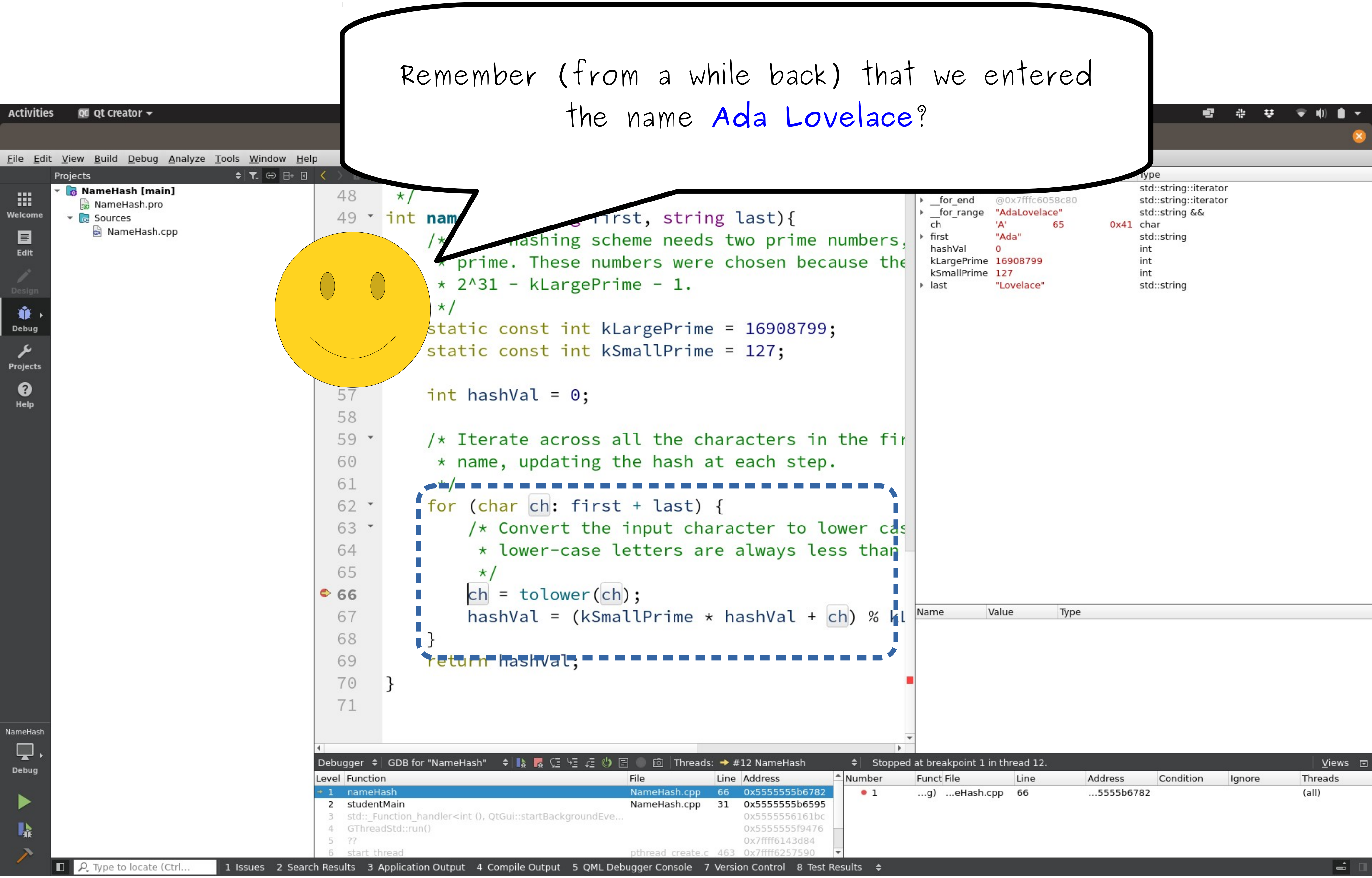
Name	Value	Type

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

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
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



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

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

				Type
				std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

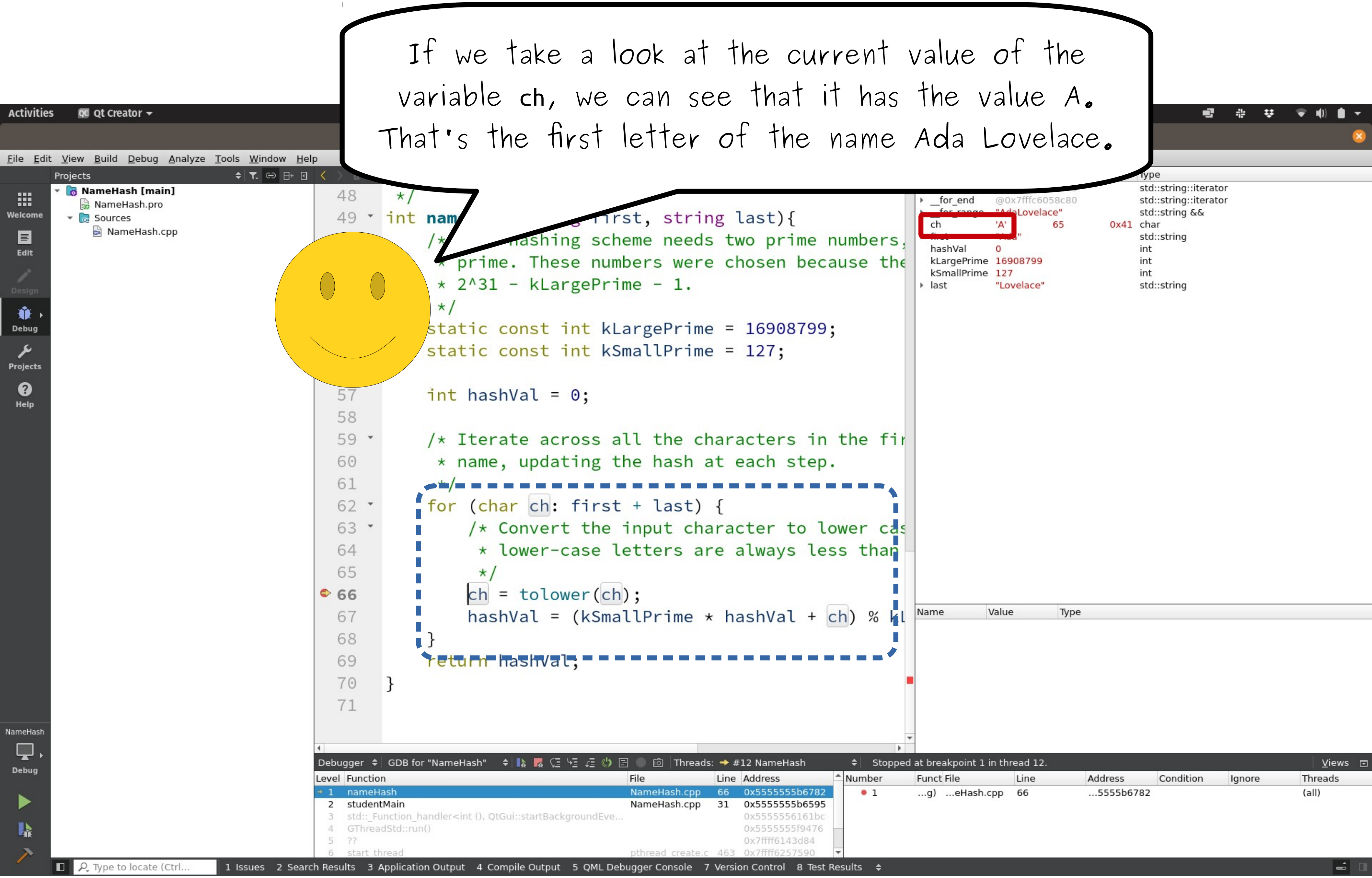
Name	Value	Type

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

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

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

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



Activities
Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results
Type to locate (Ctrl...
Views

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

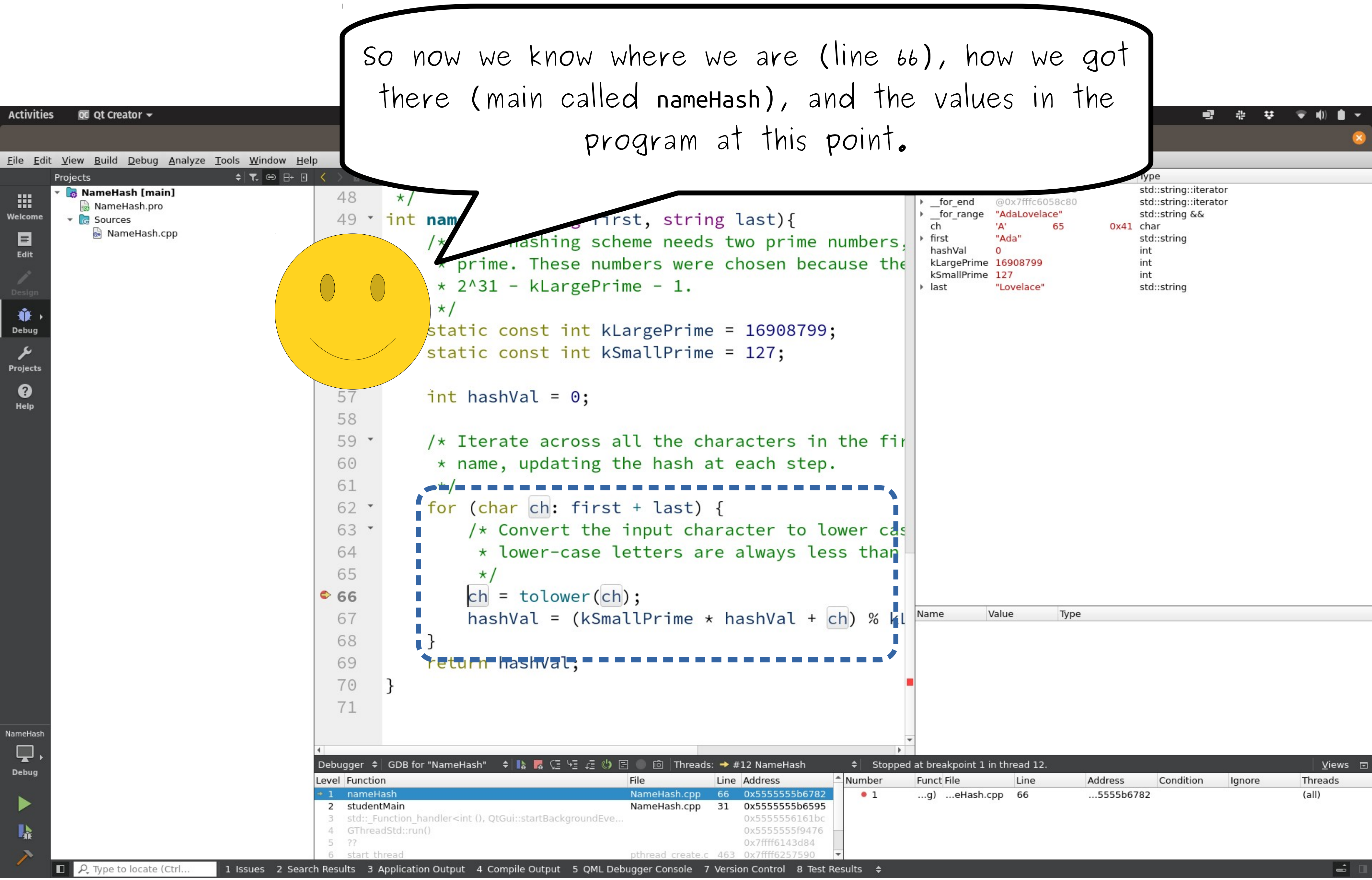
Name	Value			Type
				std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Debugger GDB for "NameHash" Threads: #12 NameHash Stopped 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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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.



```
48      */
49  int nam                    irst, string last){
        /*          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;

57      int hashVal = 0;
58
59      /* Iterate across all the characters in the fi
60       * name, updating the hash at each step.
61       */
62      for (char ch: first + last) {
63          /* Convert the input character to lower cas
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
```

Name	Value			Type
				std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

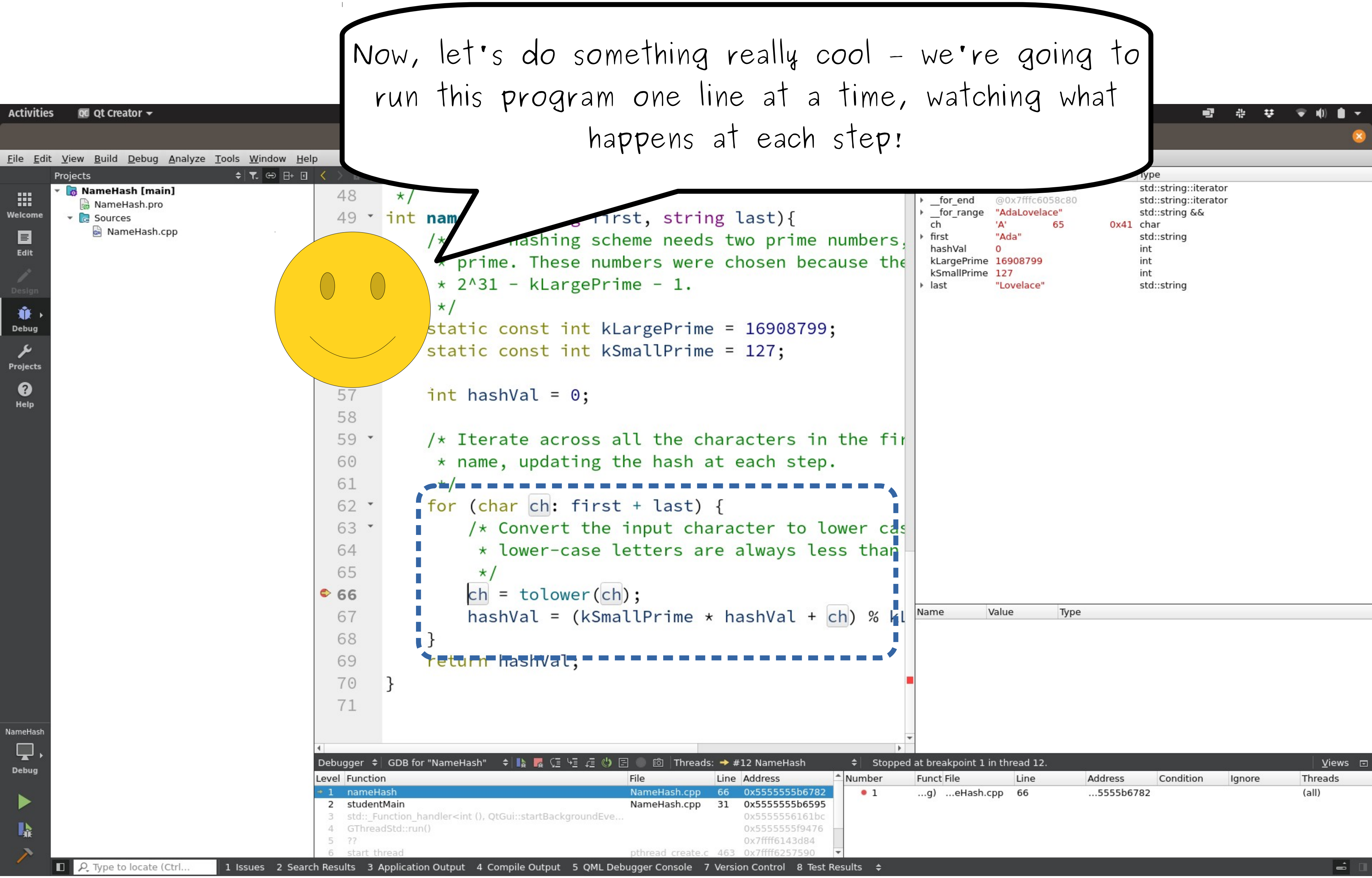
Name	Value	Type

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

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

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

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!



```
    */
int nam                first, string last){
    /*      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 fi
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kl
    }
    return hashVal;
}
```

			Type
			std::string::iterator
__for_end	@0x7fffc6058c80		std::string::iterator
__for_range	"AdaLovelace"		std::string &&
ch	'A'	65 0x41	char
first	"Ada"		std::string
hashVal	0		int
kLargePrime	16908799		int
kSmallPrime	127		int
last	"Lovelace"		std::string

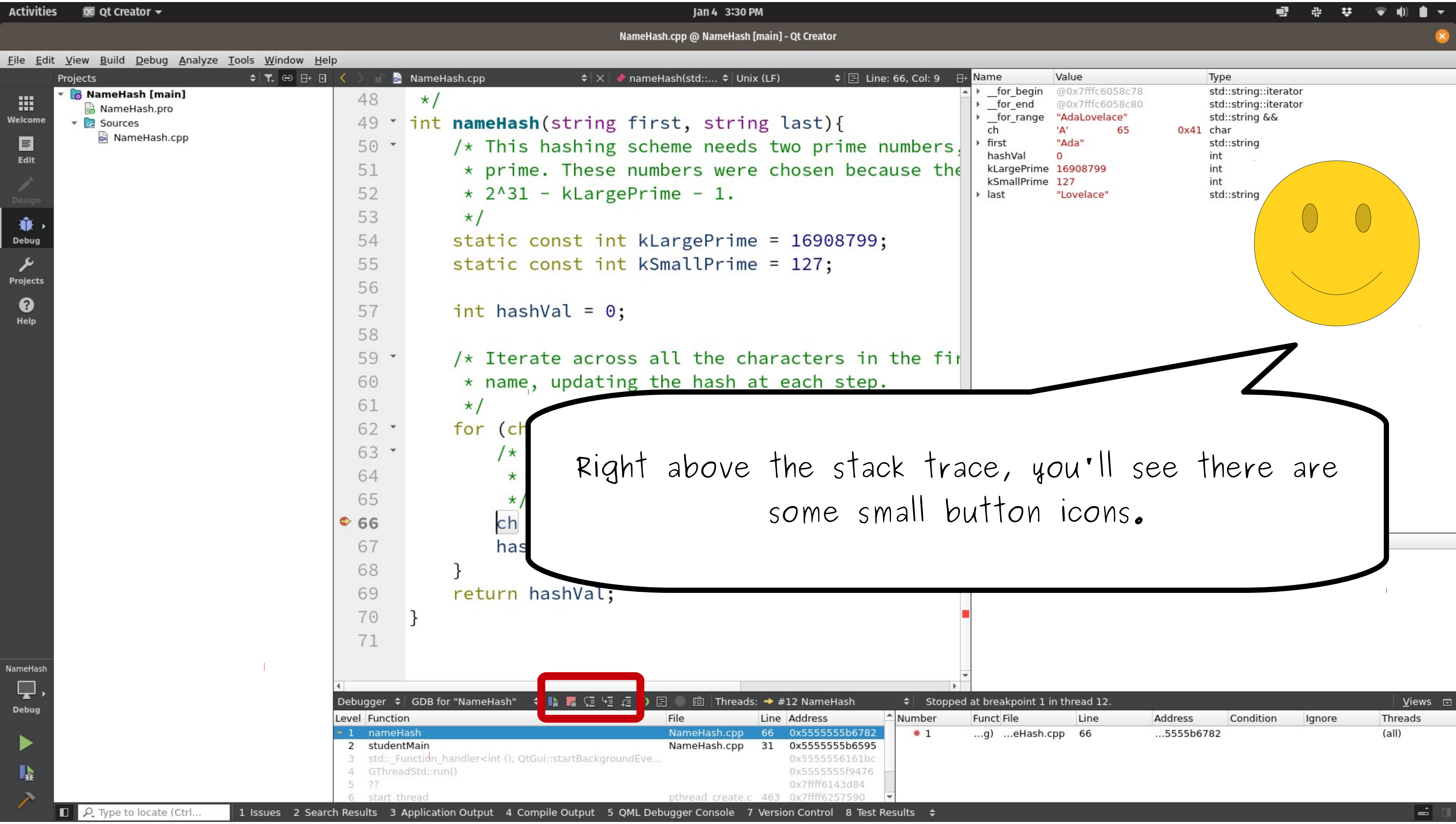
Name	Value	Type

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

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x55555555f9476
5	??			0x7ffff6143d84
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



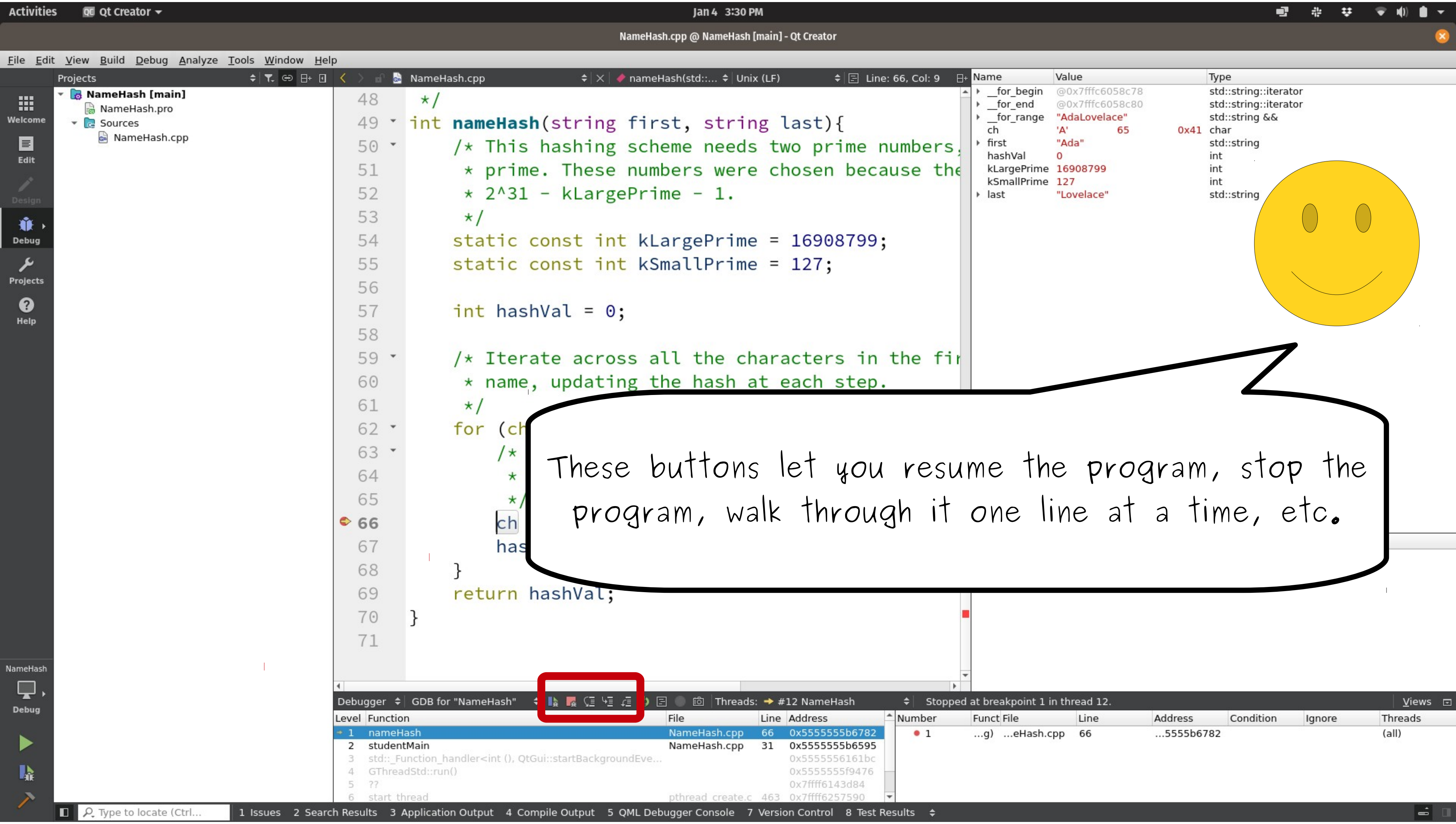
```
  */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fir
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

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

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

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



```
 */
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 fir
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

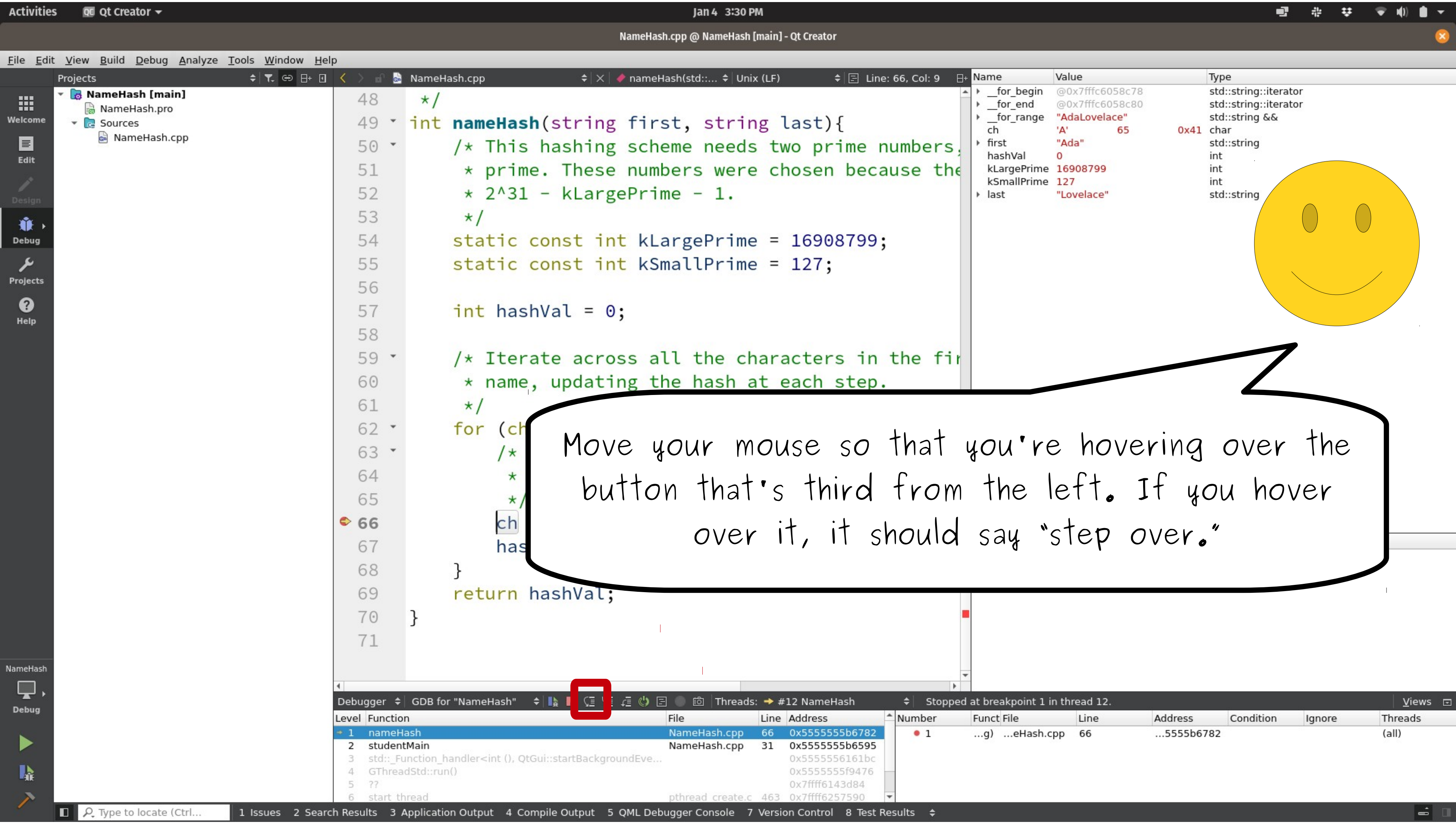
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Stopped at breakpoint 1 in thread 12.

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

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

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



```
 */
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 fir
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

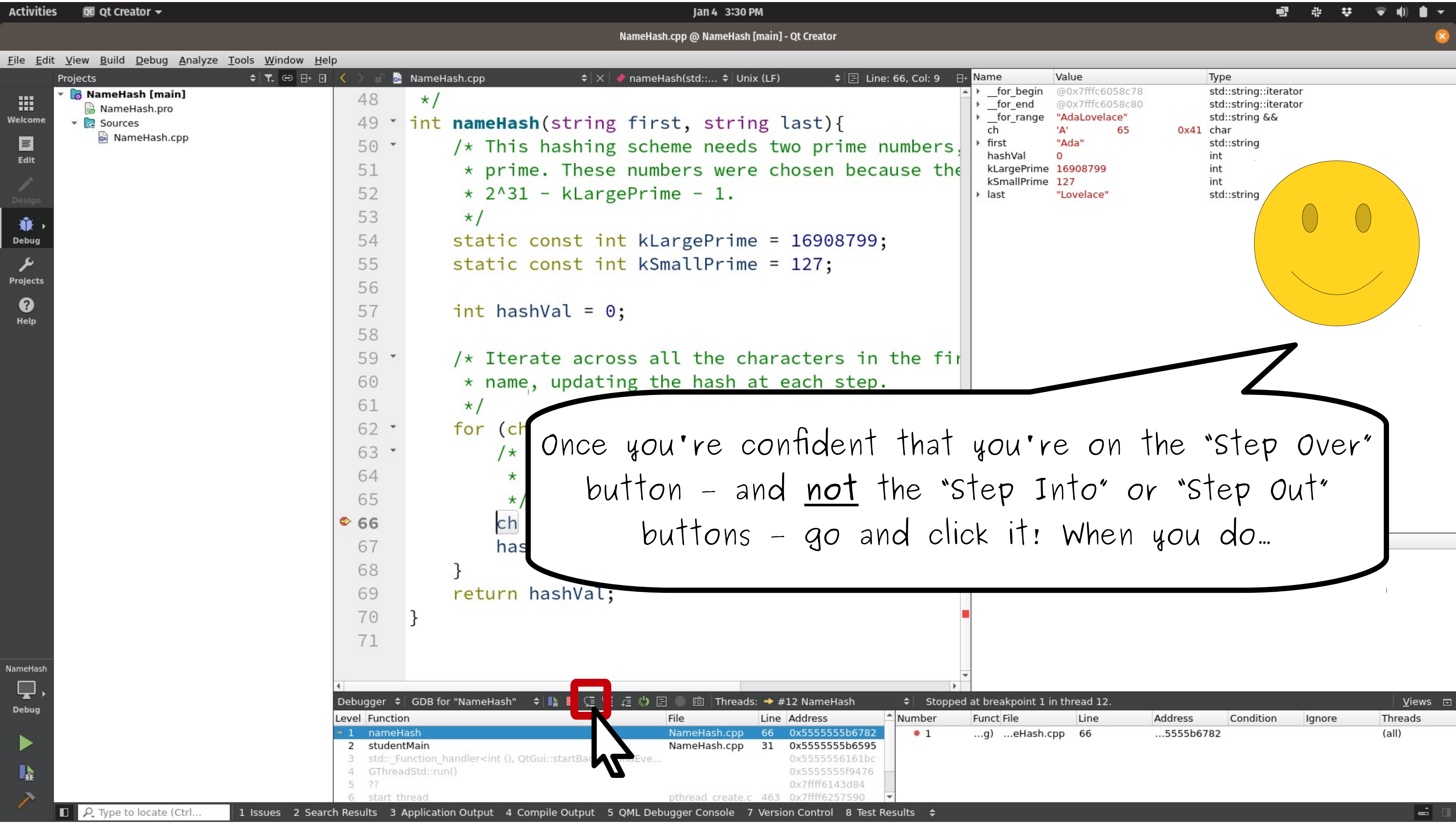
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Stopped at breakpoint 1 in thread 12.

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

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

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."



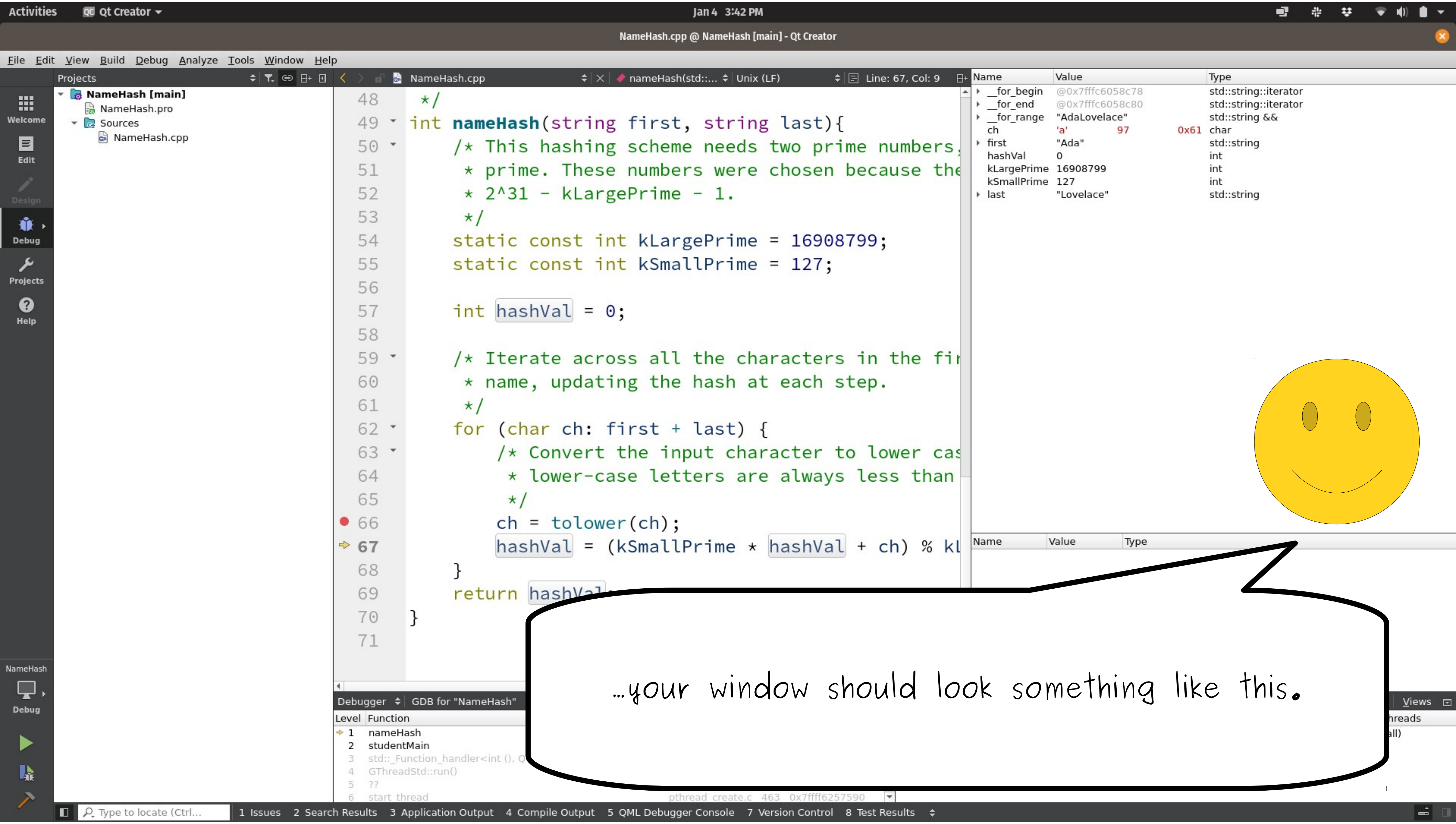
```
  */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fir
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}

```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'A'	65	0x41	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	66	0x5555555b6782
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBa…Eve...			0x555555б161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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…



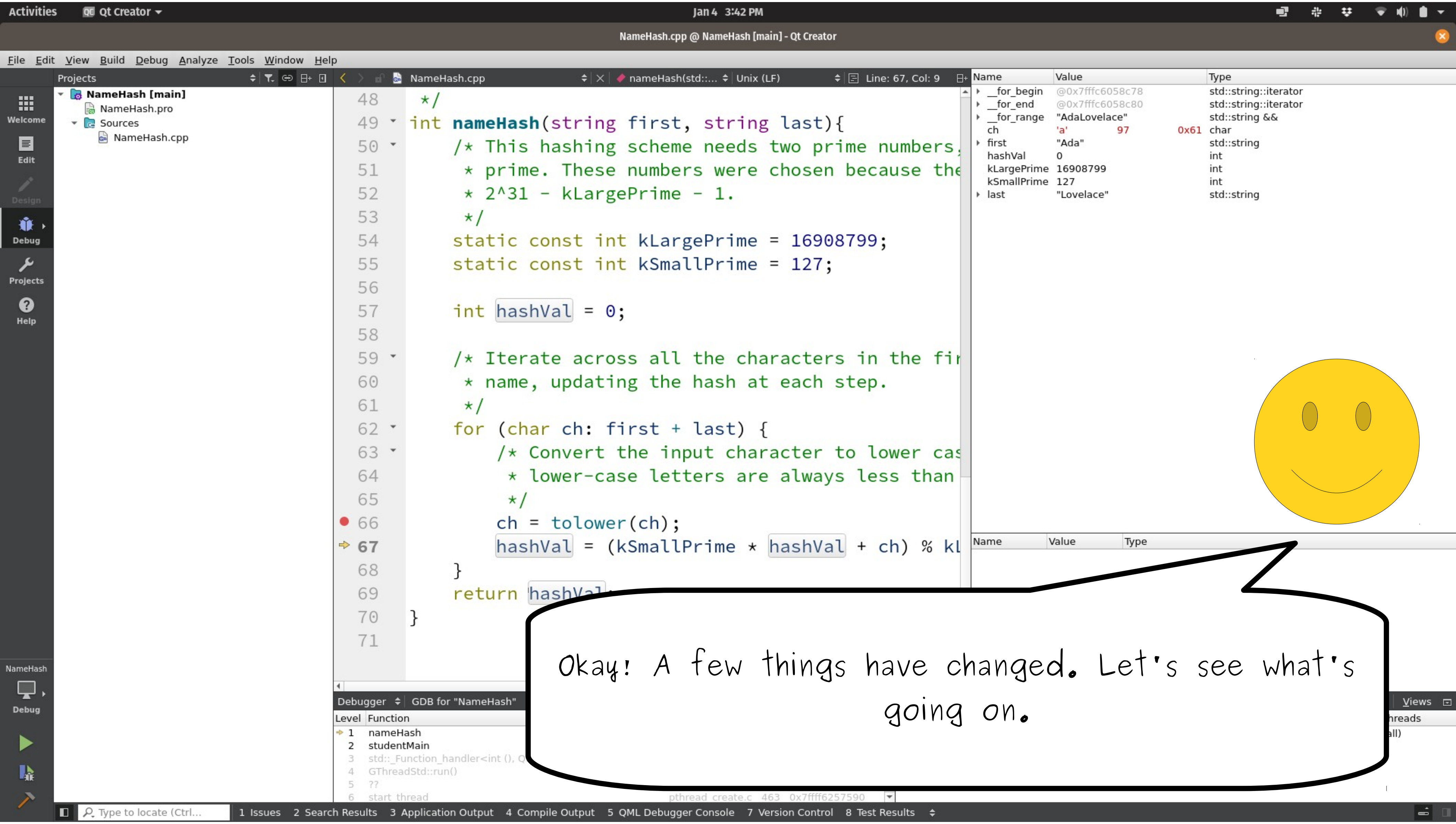
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97 0x61	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Name	Value	Type

Level	Function
1	nameHash
2	studentMain
3	std::_Function_handler<int (), Q
4	GThreadStd::run()
5	??
6	start_thread pthread_create.c 463 0x7ffff6257590

…your window should look something like this.



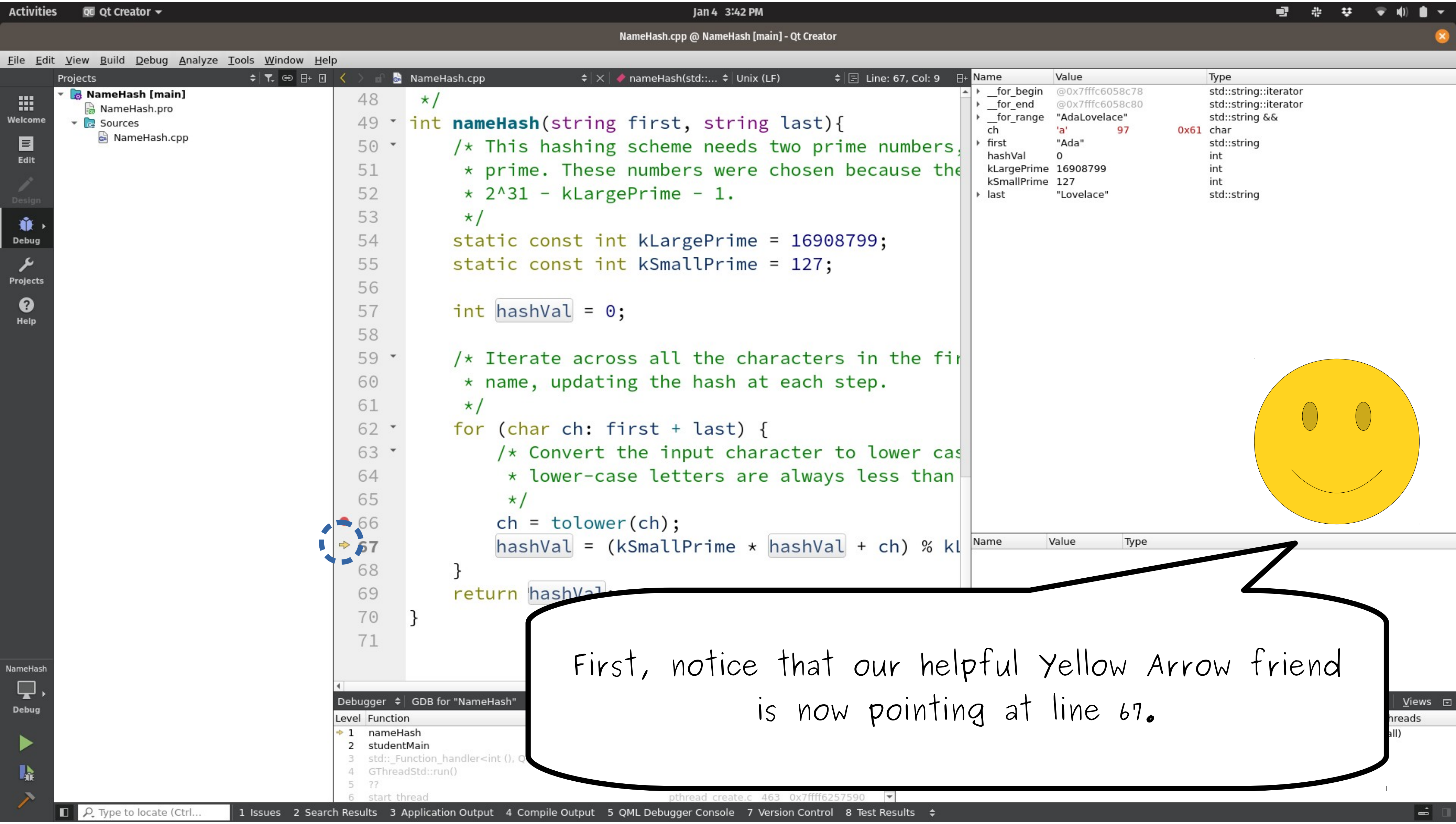
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97 0x61	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

Name	Value	Type

Level	Function
1	nameHash
2	studentMain
3	std::_Function_handler<int (), Q
4	GThreadStd::run()
5	??
6	start_thread pthread_create.c 463 0x7ffff6257590

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



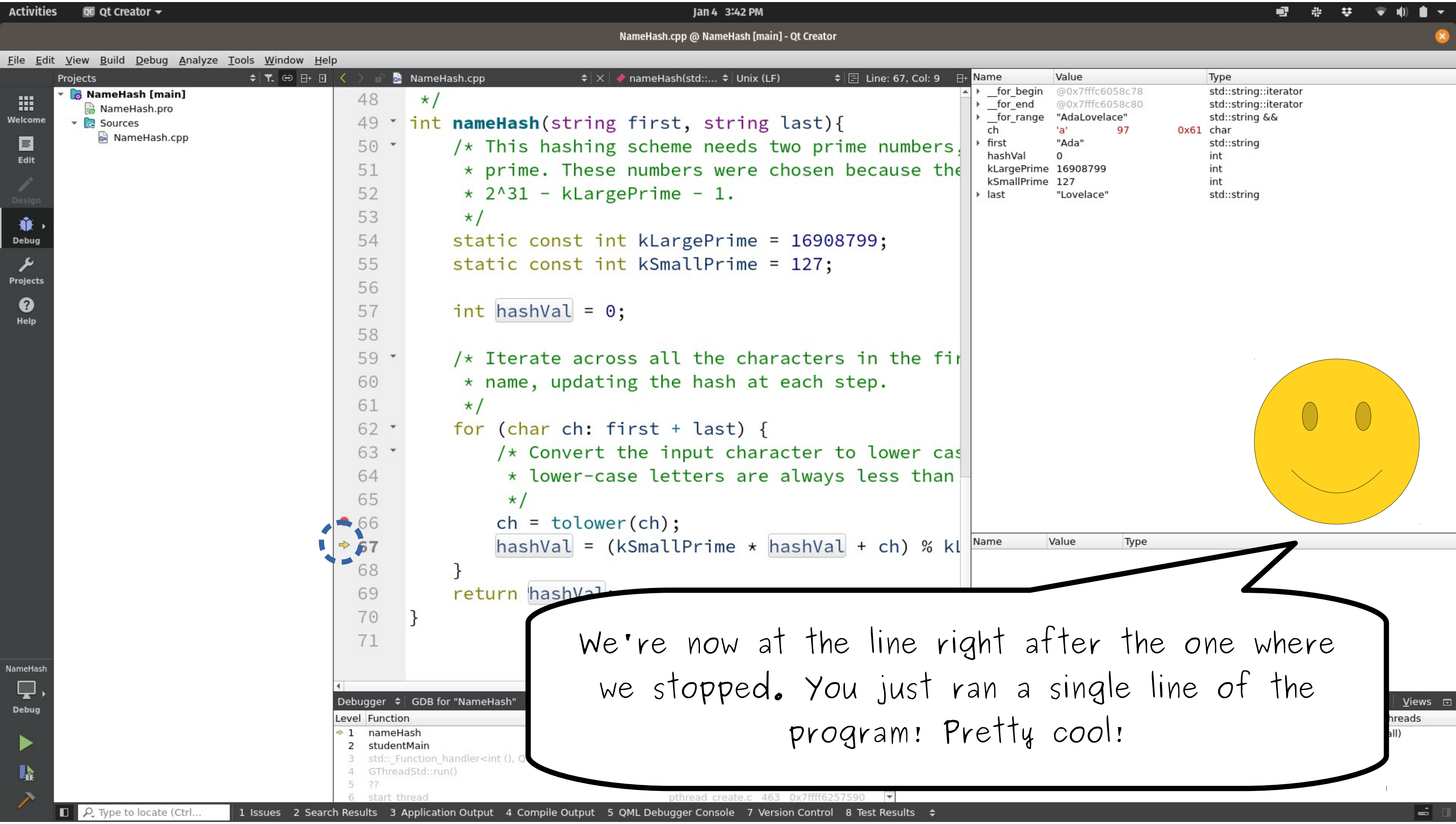
```
48      */
49  int nameHash(string first, string last){
50      /* This hashing scheme needs two prime numbers,
51       * prime. These numbers were chosen because the
52       * 2^31 - kLargePrime - 1.
53       */
54      static const int kLargePrime = 16908799;
55      static const int kSmallPrime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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

Level	Function
1	nameHash
2	studentMain
3	std::_Function_handler<int (), Q
4	GThreadStd::run()
5	??
6	start_thread pthread_create.c 463 0x7ffff6257590

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

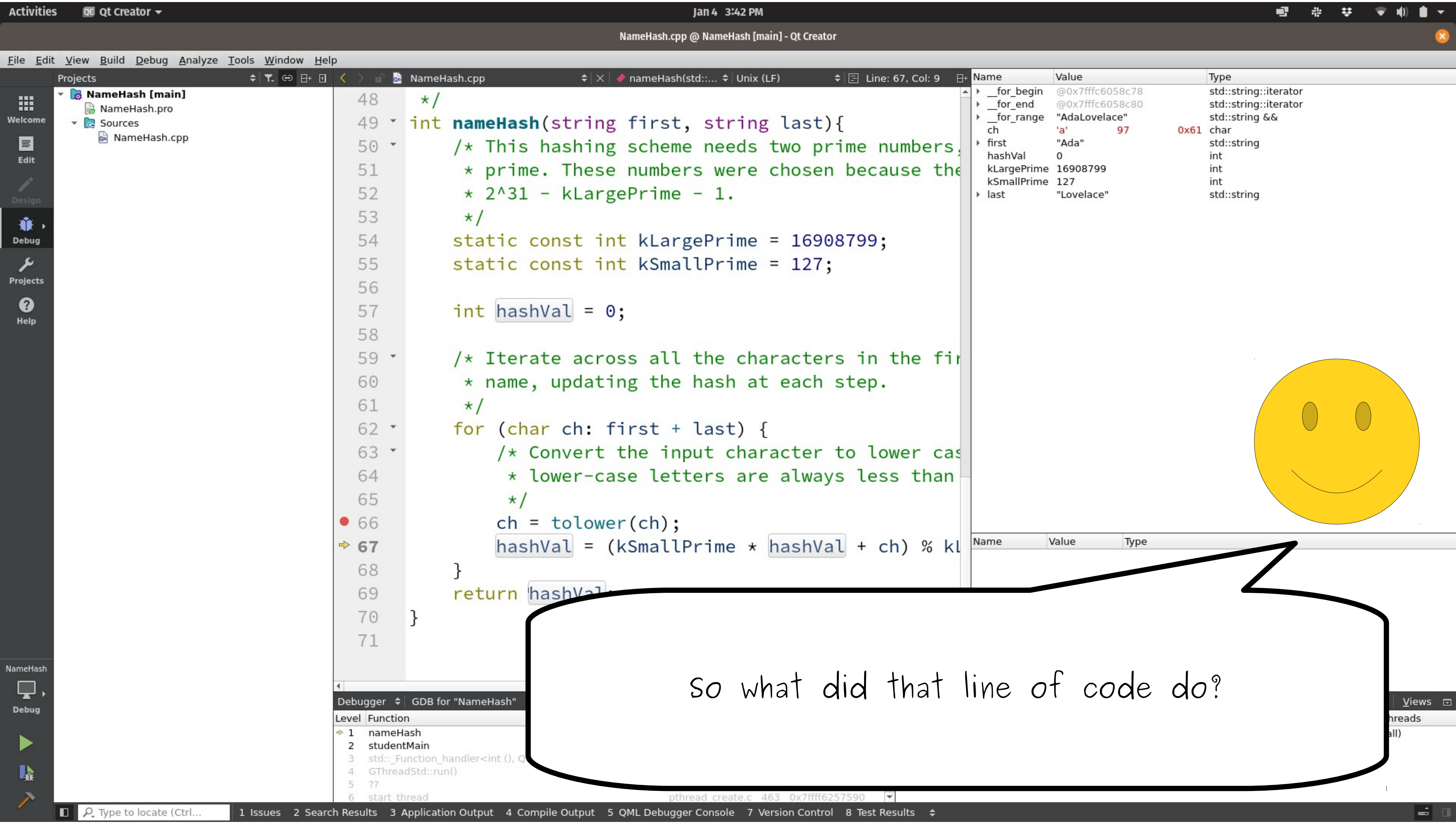


```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

```
Name            Value                              Type
__for_begin     @0x7fffc6058c78                    std::string::iterator
__for_end       @0x7fffc6058c80                    std::string::iterator
__for_range     "AdaLovelace"                      std::string &&
ch              'a'          97           0x61     char
first           "Ada"                              std::string
hashVal         0                                  int
kLargePrime     16908799                           int
kSmallPrime     127                                int
last            "Lovelace"                         std::string
```

```
Level  Function
1      nameHash
2      studentMain
3      std::_Function_handler<int (), Q
4      GThreadStd::run()
5      ??
6      start_thread                      pthread_create.c   463   0x7ffff6257590
```

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

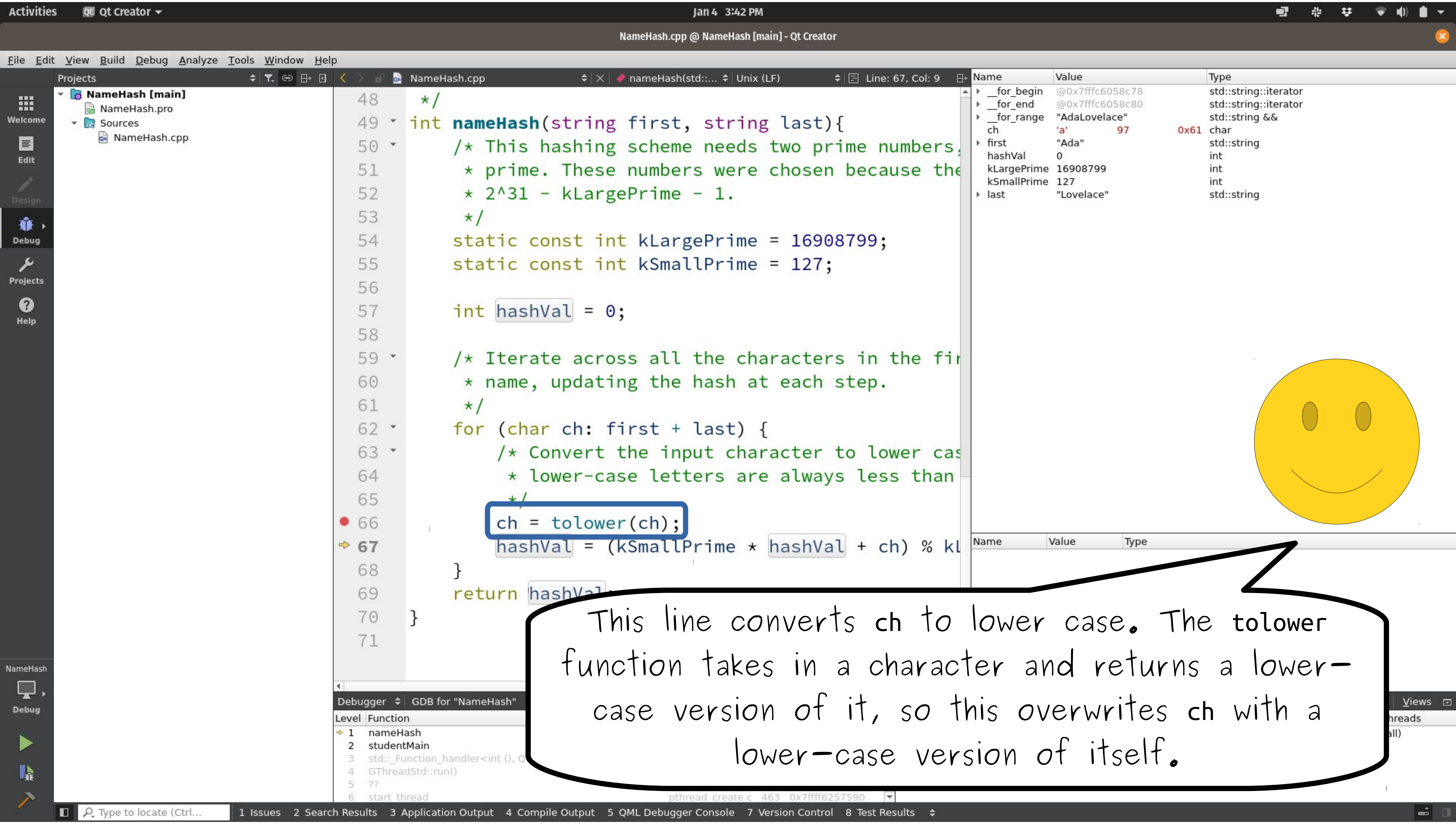


```
    */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97 0x61	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

```
Level Function
1     nameHash
2     studentMain
3     std::_Function_handler<int (), Q
4     GThreadStd::run()
5     ??
6     start_thread                        pthread_create.c  463  0x7ffff6257590
```

So what did that line of code do?



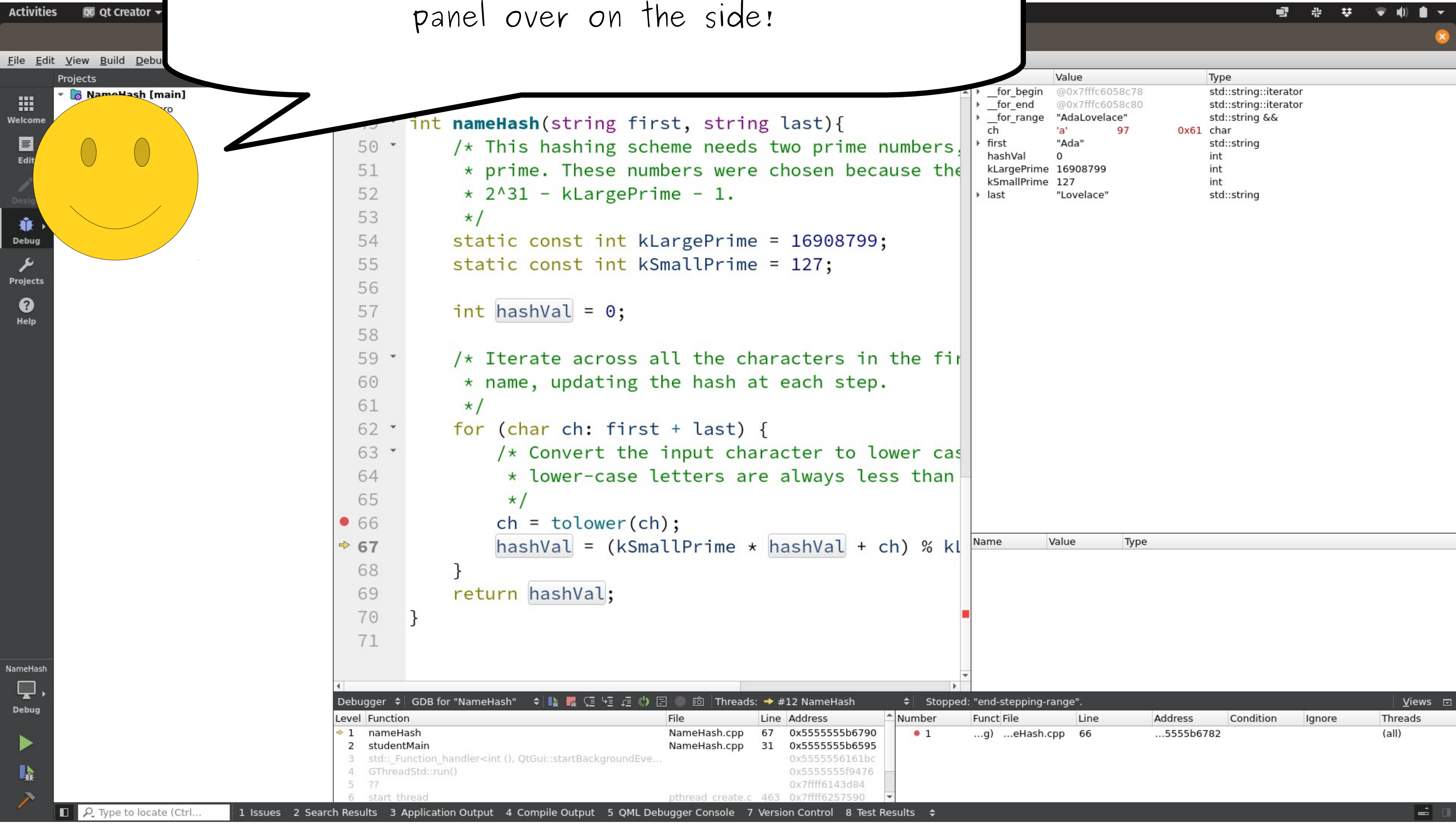
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Level	Function
1	nameHash
2	studentMain
3	std::_Function_handler<int (), Q
4	GThreadStd::run()
5	??
6	start_thread pthread_create.c 463 0x7ffff6257590

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!


Activities
Qt Creator
File Edit View Build Debu
Projects
NameHash [main]
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Debugger
GDB for "NameHash"
Threads: #12 NameHash
Stopped: "end-stepping-range".
Views
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

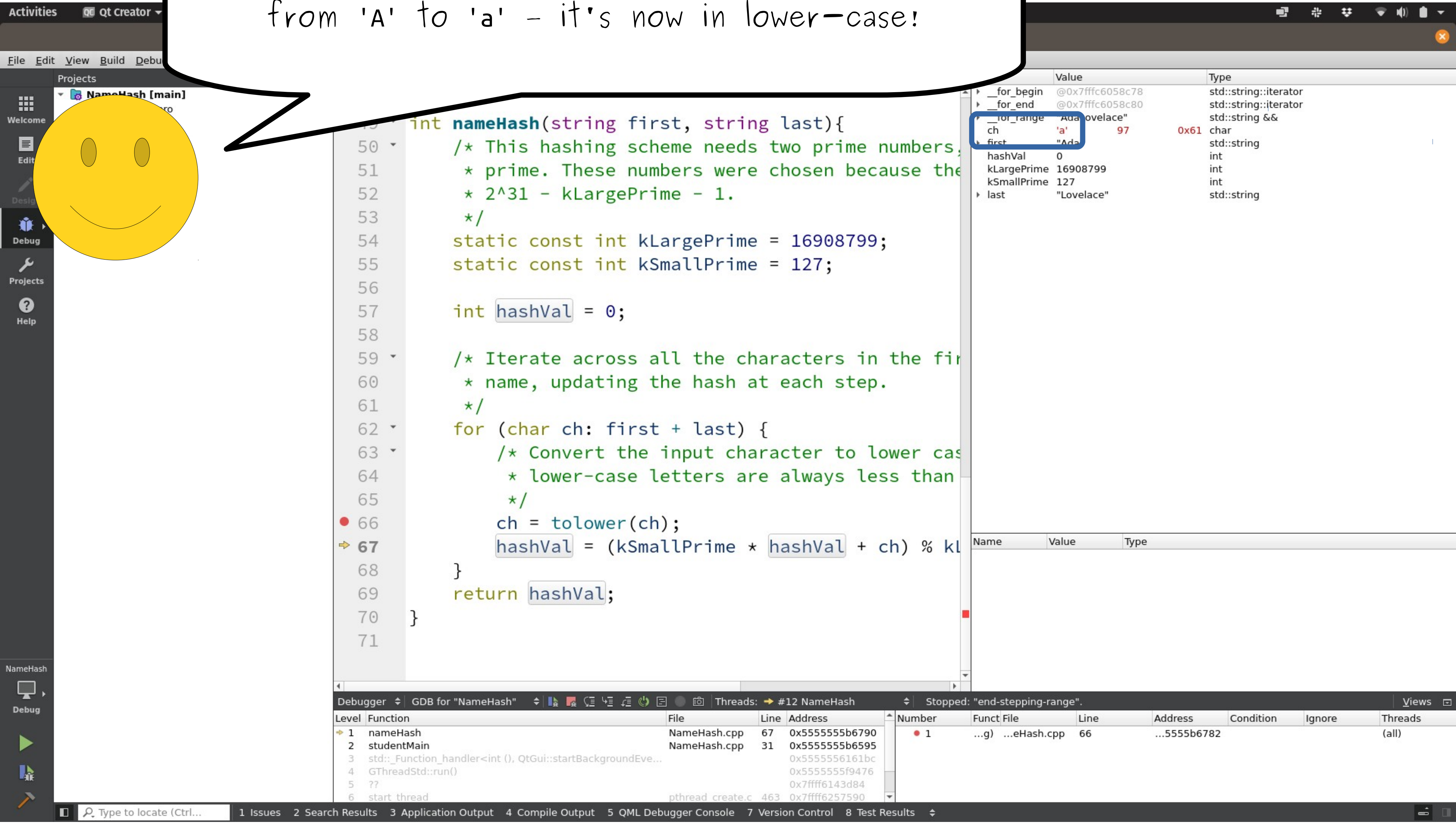
```
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

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

```
Name           Value                       Type
__for_begin    @0x7fffc6058c78             std::string::iterator
__for_end      @0x7fffc6058c80             std::string::iterator
__for_range    "AdaLovelace"               std::string &&
ch             'a'        97        0x61   char
first          "Ada"                       std::string
hashVal        0                           int
kLargePrime    16908799                    int
kSmallPrime    127                         int
last           "Lovelace"                  std::string
```

```
Level  Function                                              File              Line  Address
1      nameHash                                              NameHash.cpp      67    0x5555555b6790
2      studentMain                                           NameHash.cpp      31    0x5555555b6595
3      std::_Function_handler<int (), QtGui::startBackgroundEve...                   0x5555556161bc
4      GThreadStd::run()                                                             0x5555555f9476
5      ??                                                                            0x7ffff6143d84
6      start_thread                                          pthread_create.c  463   0x7ffff6257590
```

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

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



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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"Ada Lovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

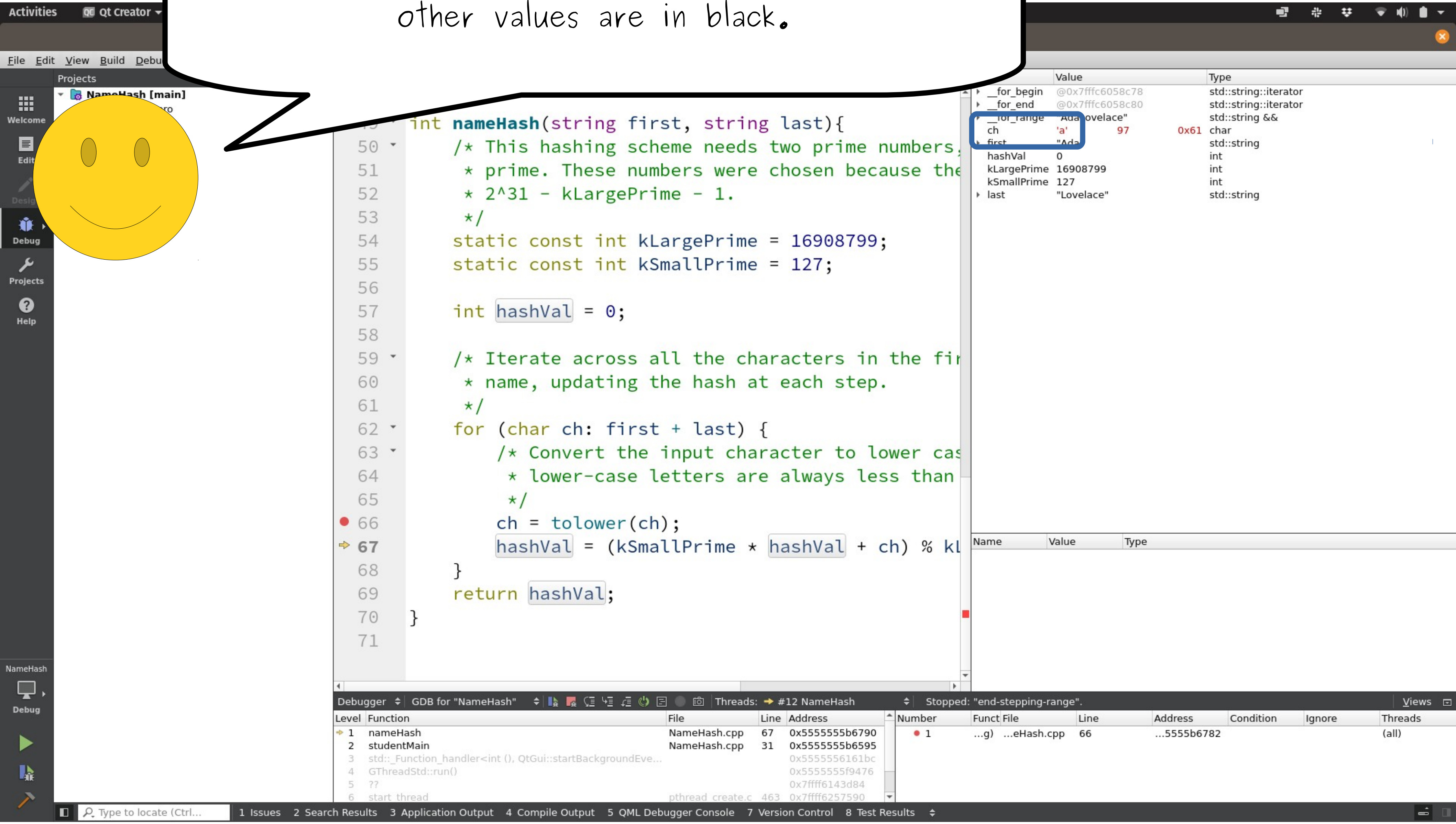
Debugger GDB for "NameHash" Threads: #12 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

Activities
Qt Creator
File Edit View Build Debu
Projects
NameHash [main]
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Views
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 fi
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower cas
         * lower-case letters are always less than
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kL
    }
    return hashVal;
}
```

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"Ada Lovelace"	std::string &&
ch	'a' 97 0x61	char
first	"Ada	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

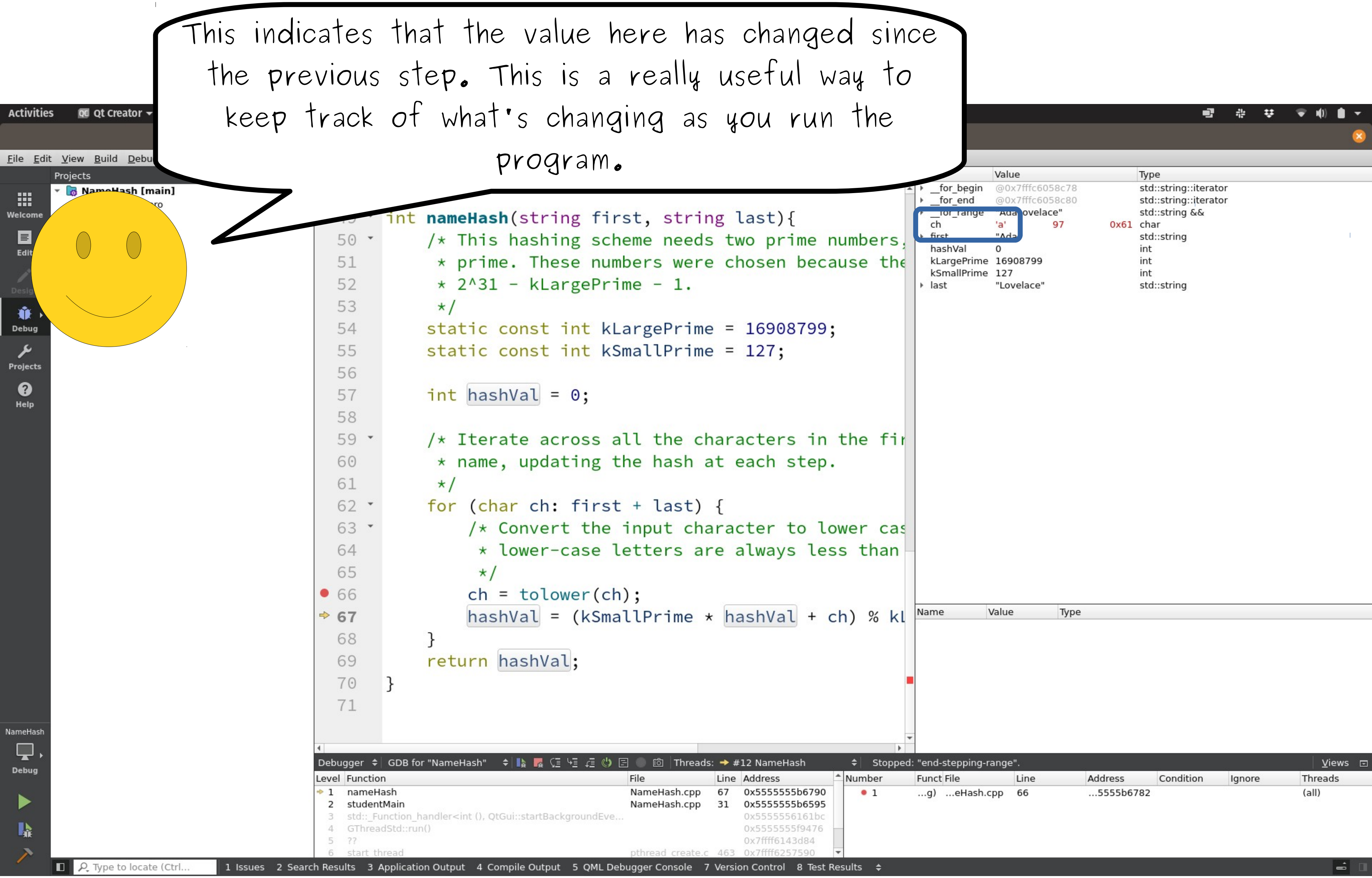
Name	Value	Type

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

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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

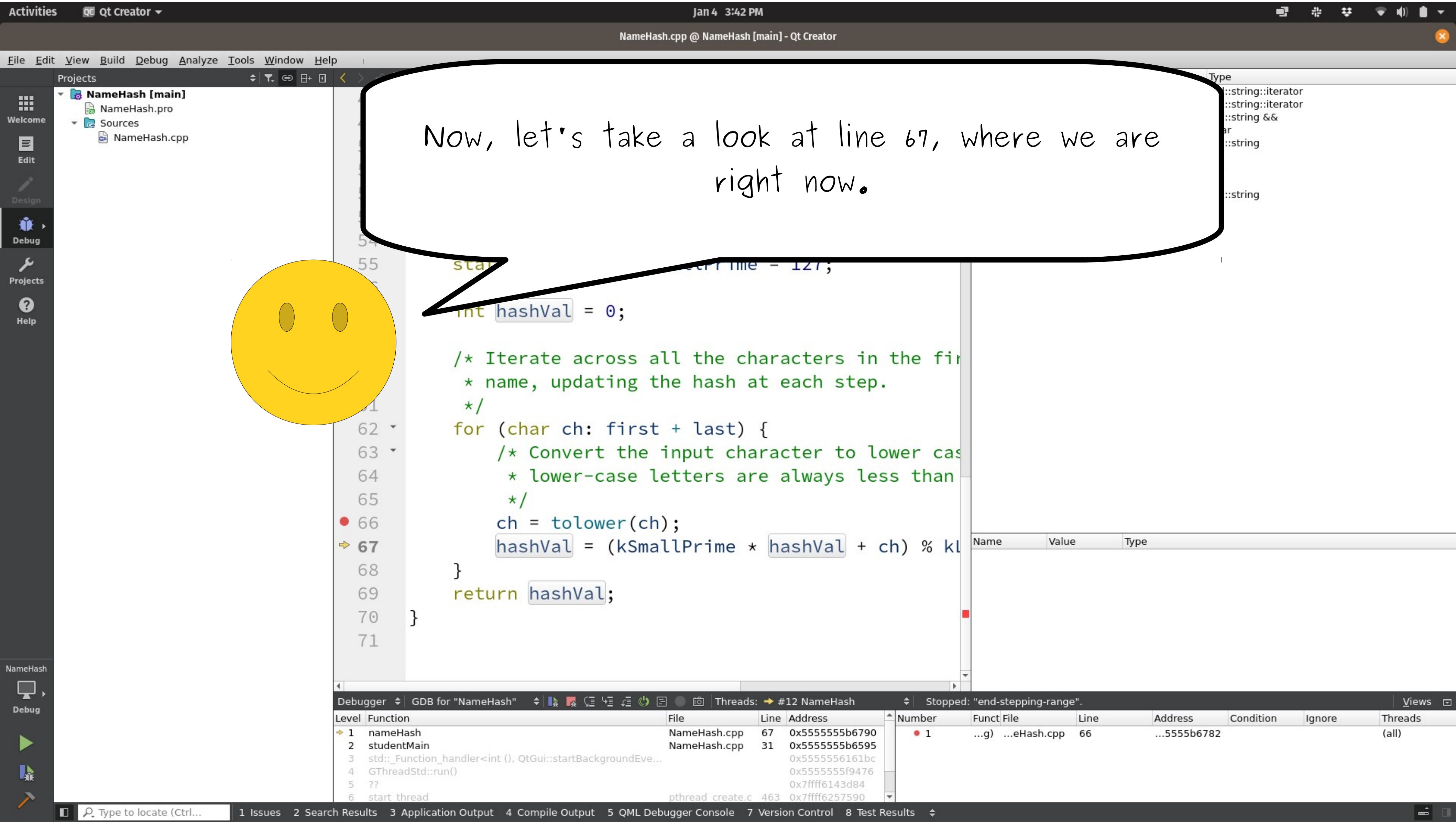
	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"Ada Lovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



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

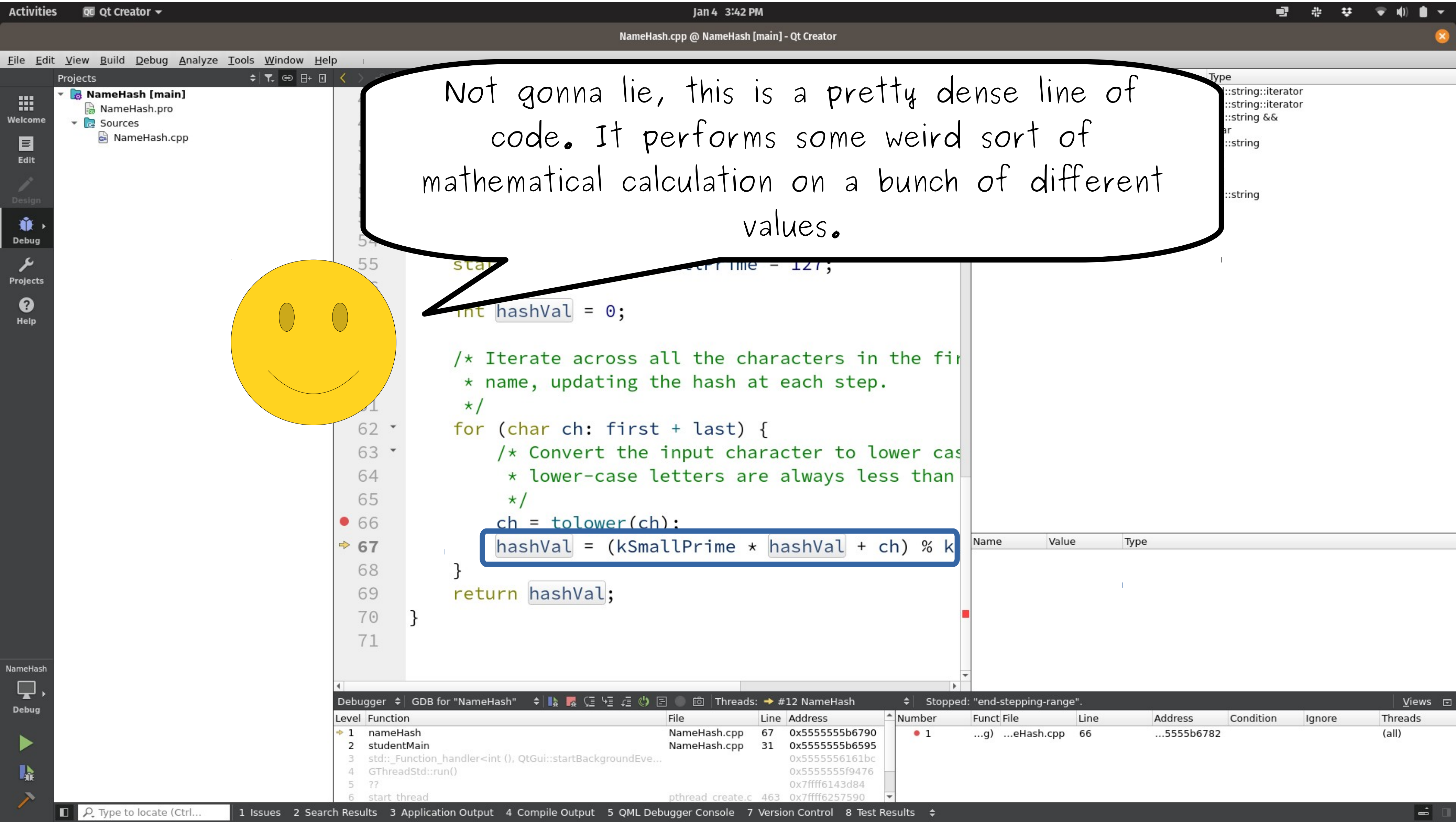
```
54
55      sta            rPrime = 127;

        int hashVal = 0;

        /* Iterate across all the characters in the fi
         * name, updating the hash at each step.
         */
62      for (char ch: first + last) {
63          /* Convert the input character to lower cas
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
```

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

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve…			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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.



```
    sta                    rPrime = 127;

    int hashVal = 0;

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

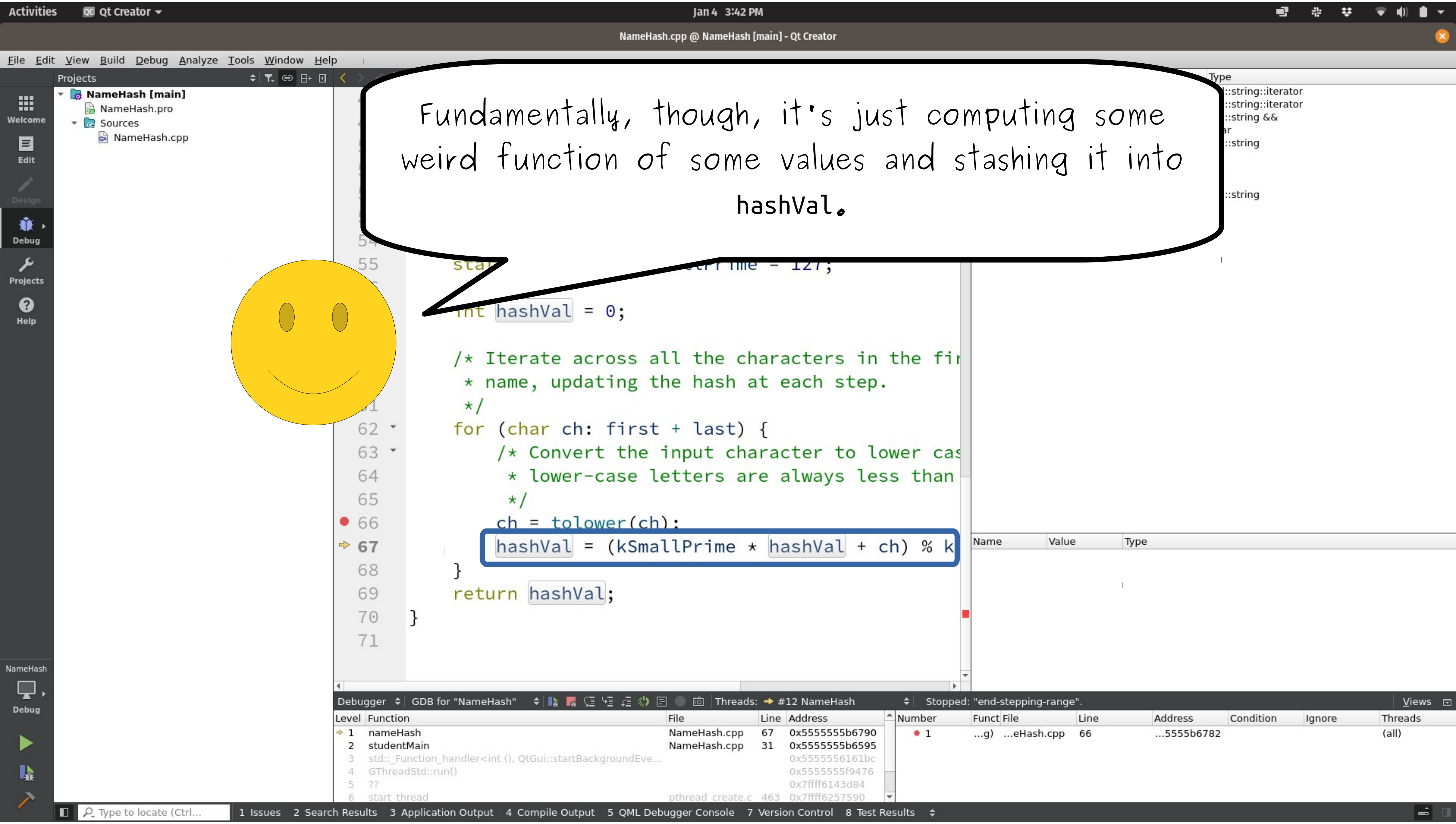
Debugger | GDB for "NameHash" | Threads: #12 NameHash | Stopped: "end-stepping-range". | Views

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
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

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

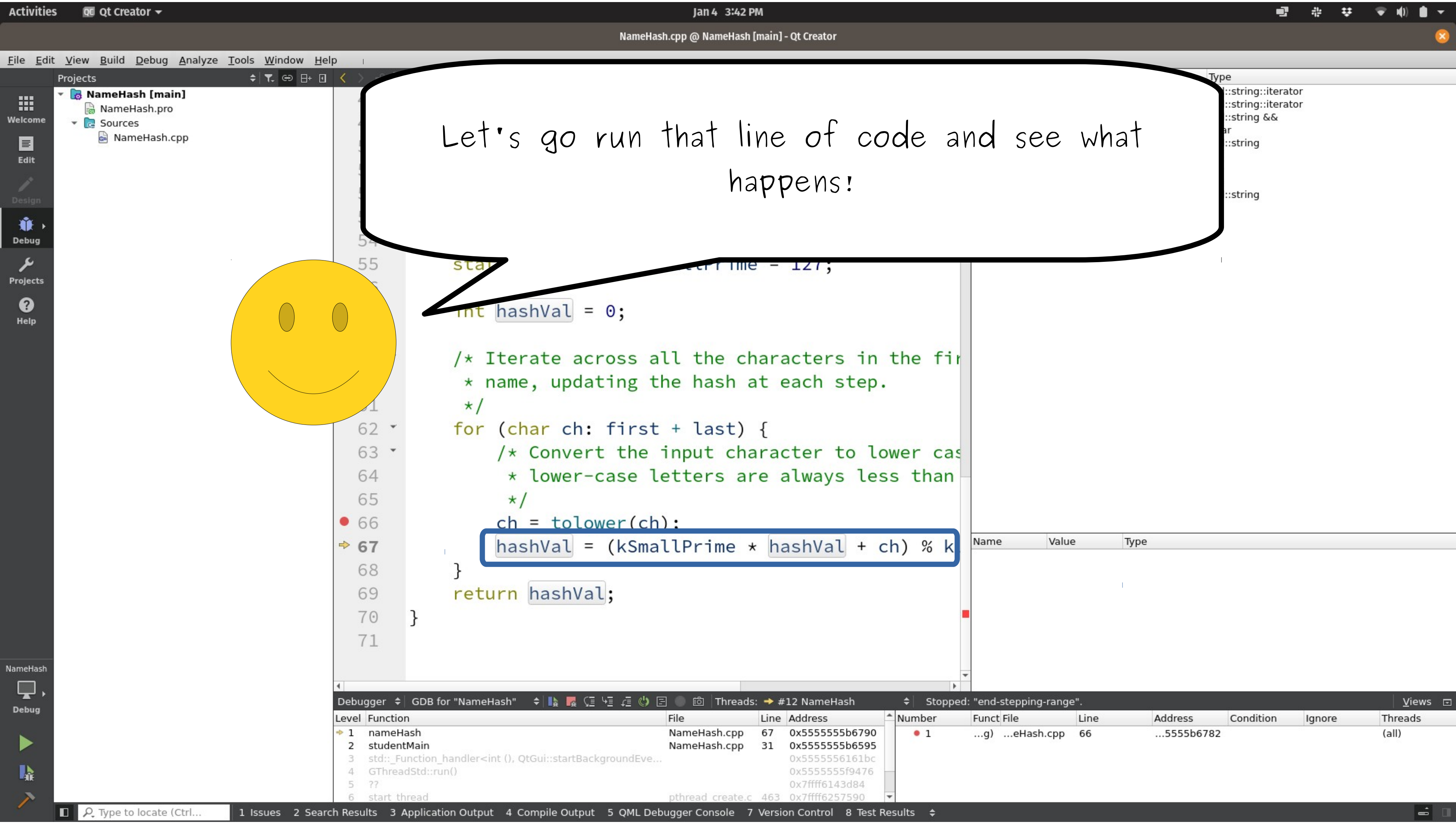


```
    sta                 rPrime = 127;

    int hashVal = 0;

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

Stopped: "end-stepping-range".

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

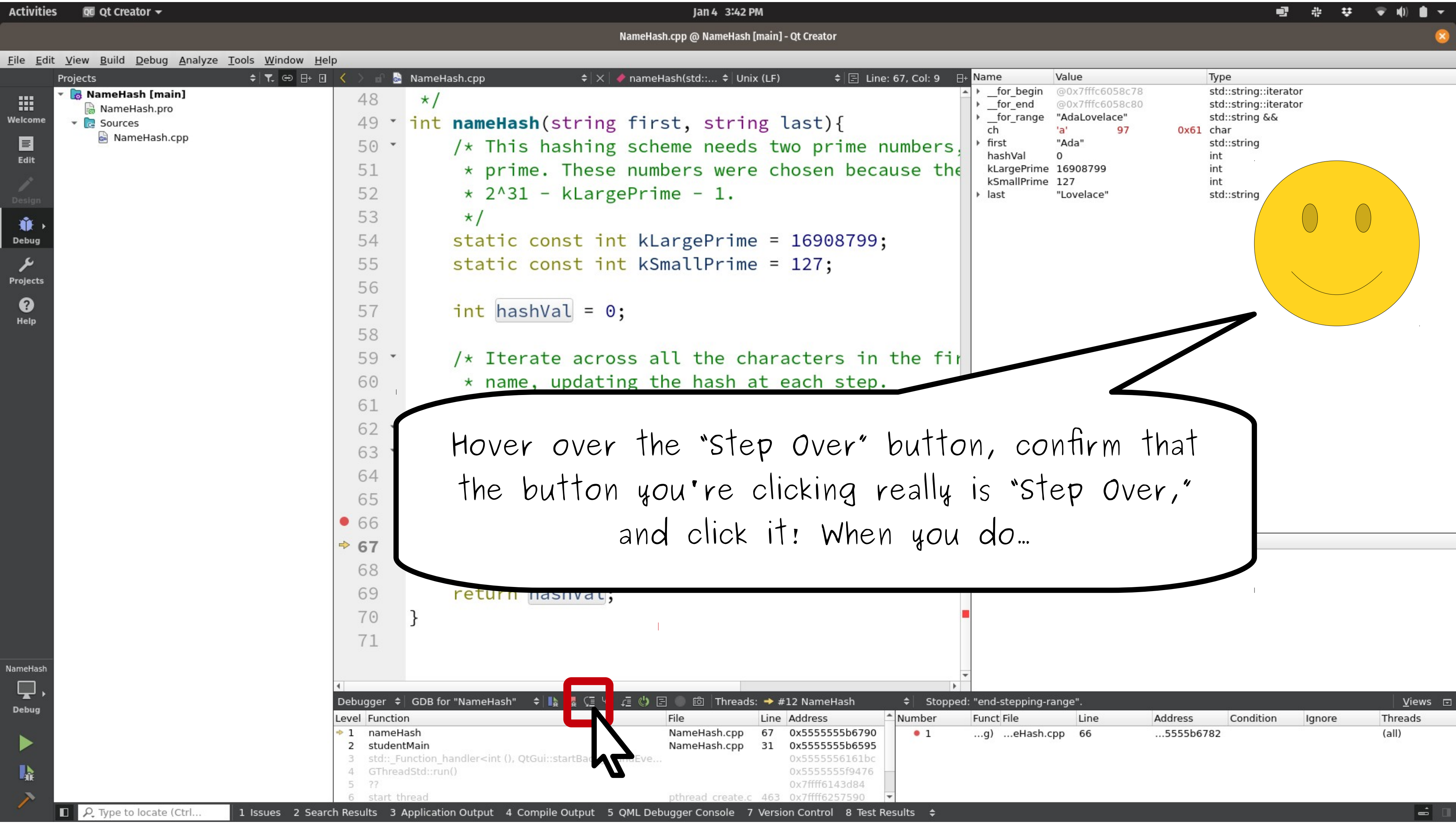


Let's go run that line of code and see what happens!

```
    sta                 ime = 127;

    int hashVal = 0;

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

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



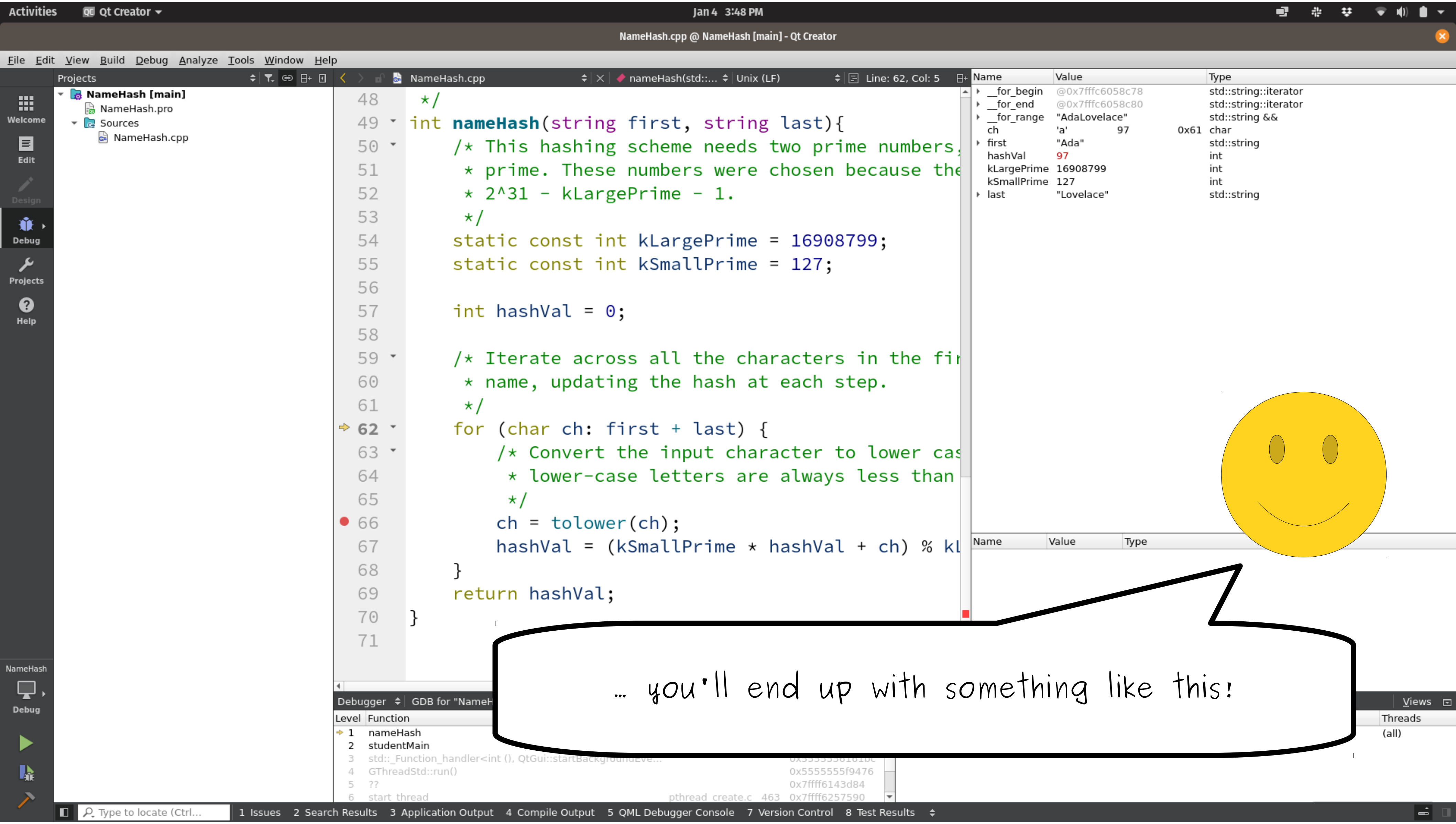
```
  */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fi
     * name, updating the hash at each step.

    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	0			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBa...dEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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



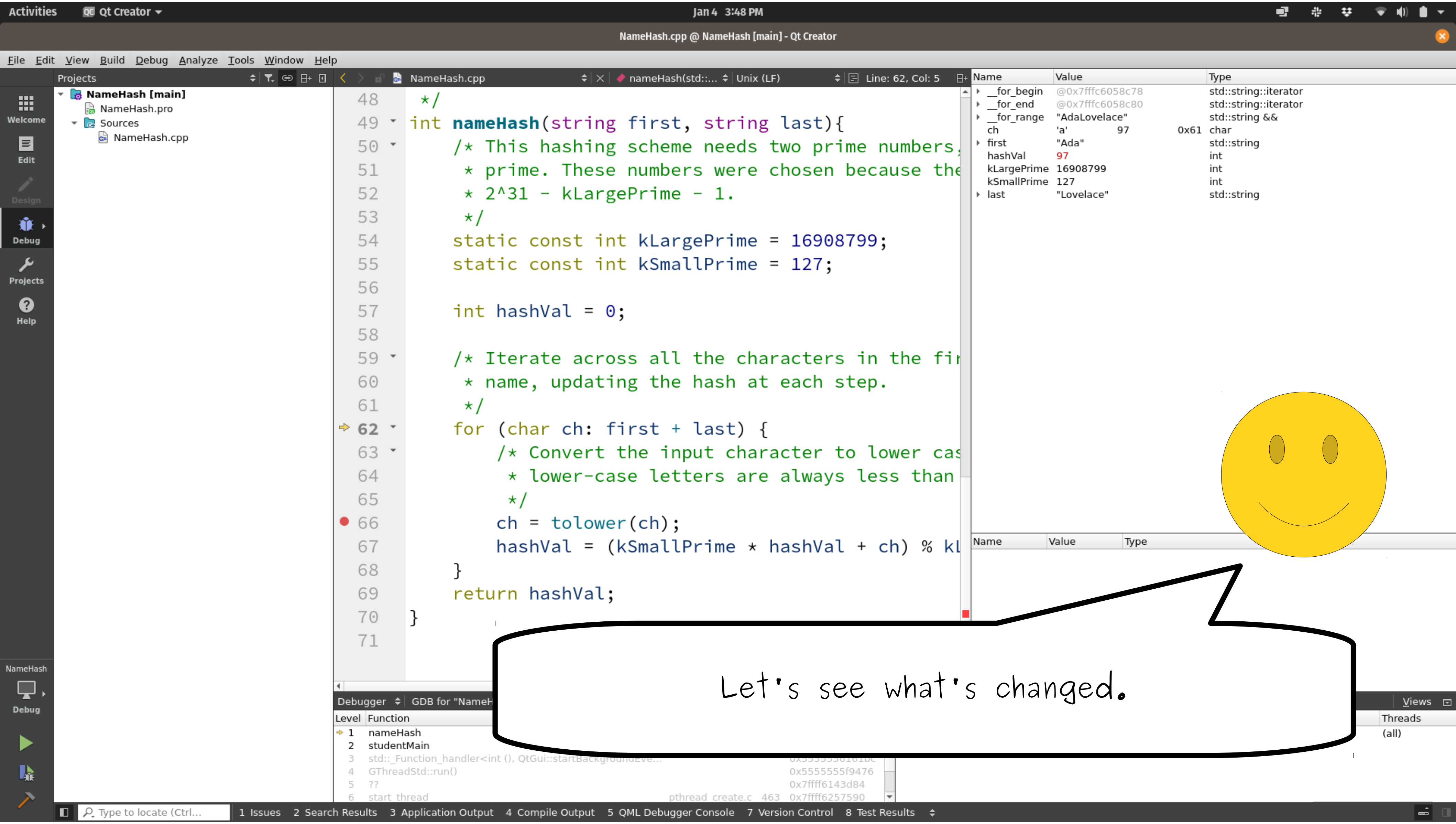
Activities
Qt Creator
Jan 4 3:48 PM
NameHash.cpp @ NameHash [main] - Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
nameHash(std::...
Unix (LF)
Line: 62, Col: 5
Debugger
GDB for "NameH
Views
Threads
(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
… you'll end up with something like this!

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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	97			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

Level	Function			
1	nameHash			
2	studentMain			
3	std::_Function_handler<int (), QtGui::startBackgroundEve...			0x5555555610bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590



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

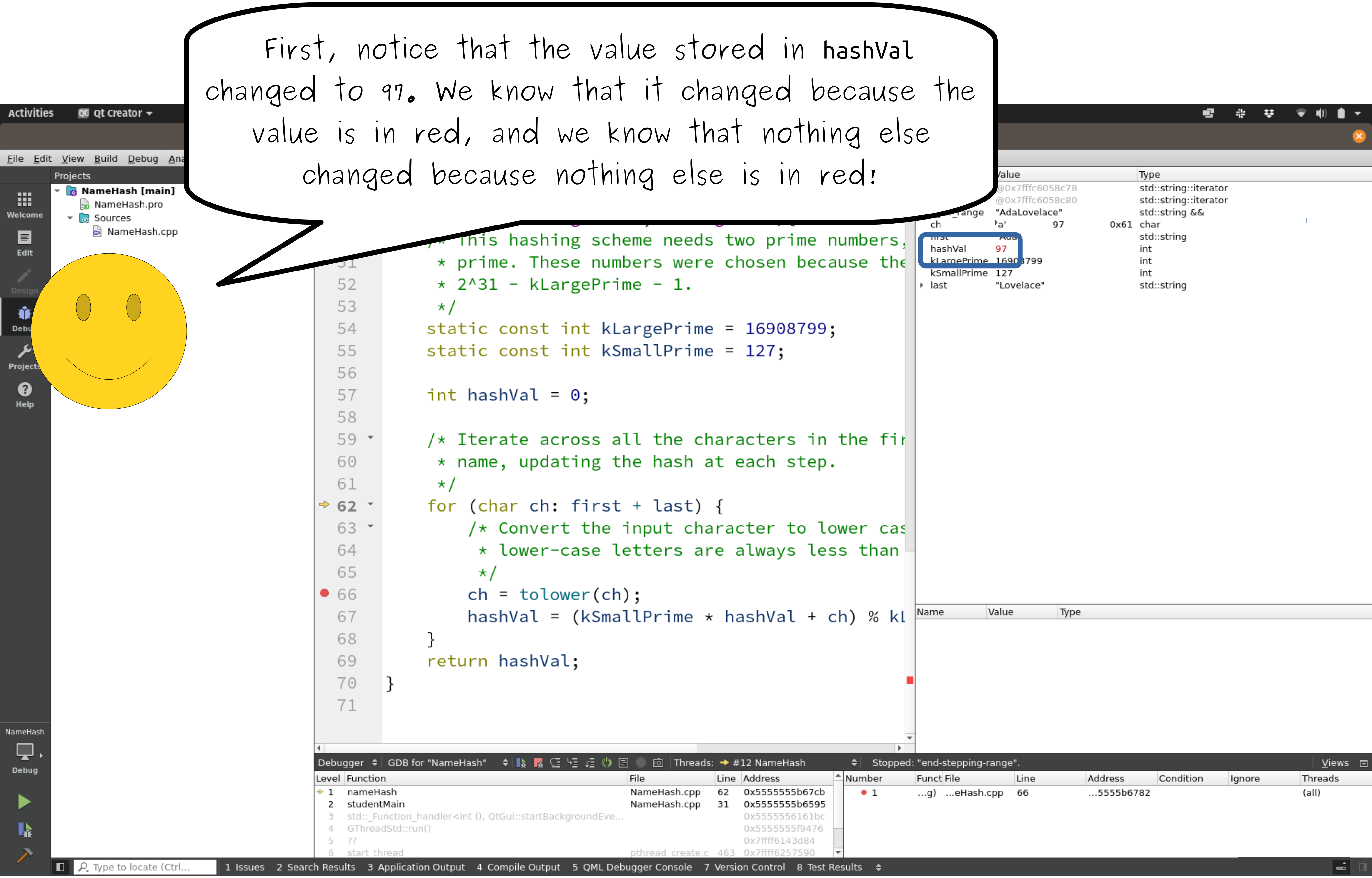
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	97			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

```
Level Function
1     nameHash
2     studentMain
3     std::_Function_handler<int (), QtGui::startBackgroundEve...
4     GThreadStd::run()                                              0x5555555f9476
5     ??                                                             0x7ffff6143d84
6     start_thread                           pthread_create.c   463  0x7ffff6257590
```

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!



Activities
Qt Creator
File Edit View Build Debug Ana
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
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

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

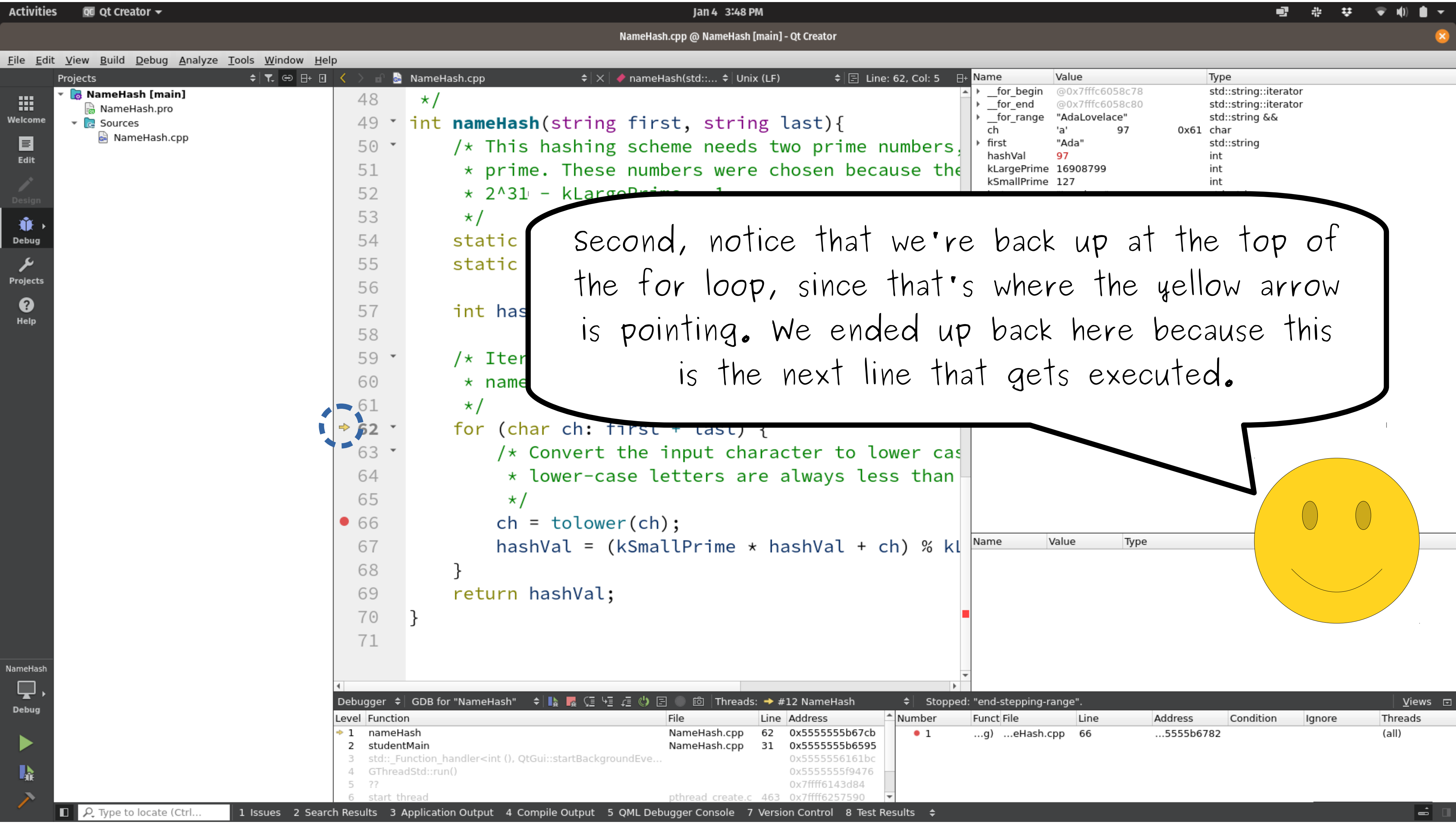
	Value			Type
	@0x7fffc6058c78			std::string::iterator
	@0x7fffc6058c80			std::string::iterator
_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	97			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



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.

```
  */
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
    static

    int has

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

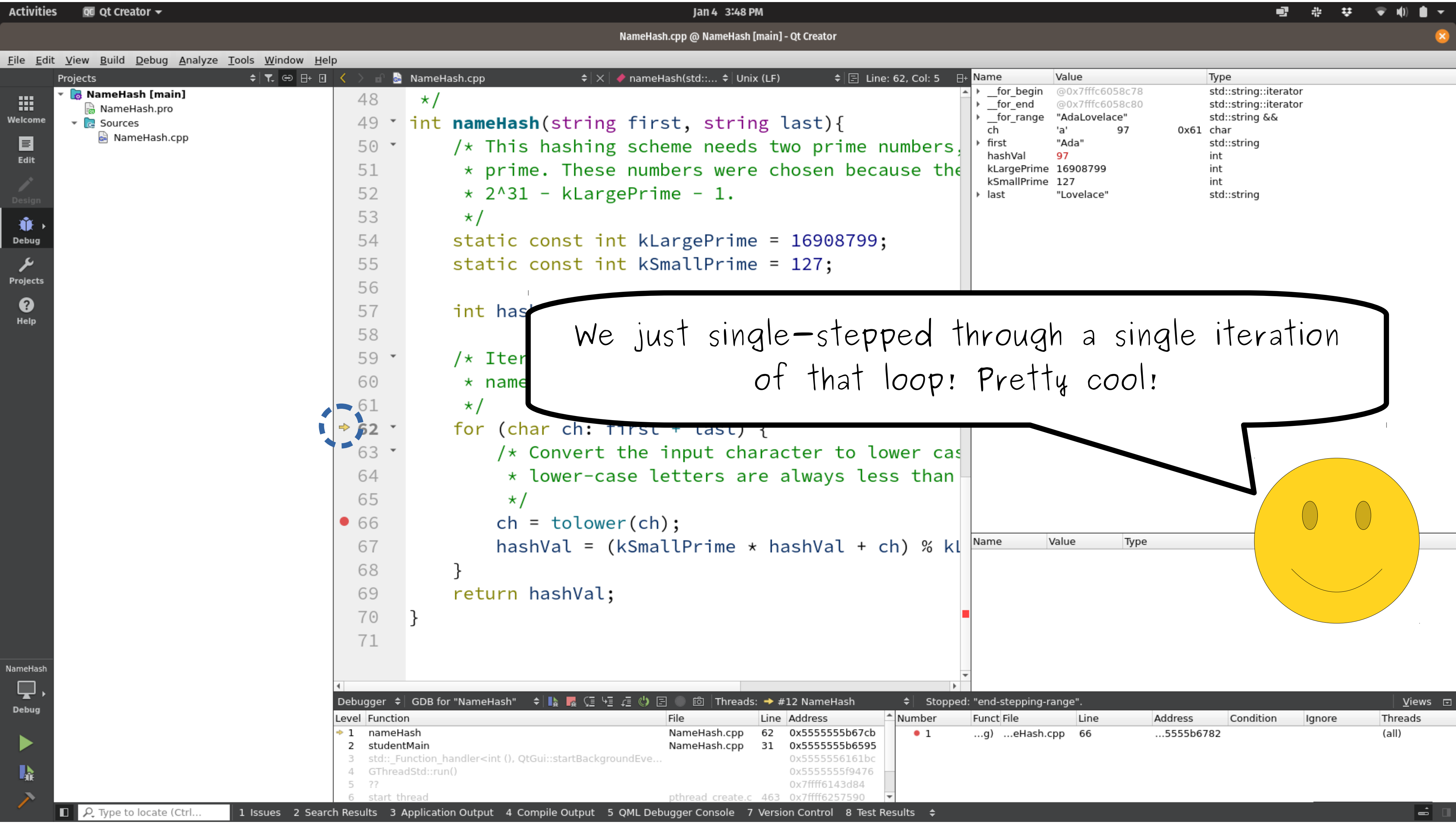
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	97			int
kLargePrime	16908799			int
kSmallPrime	127			int

Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



```
 */
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 has

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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	97			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

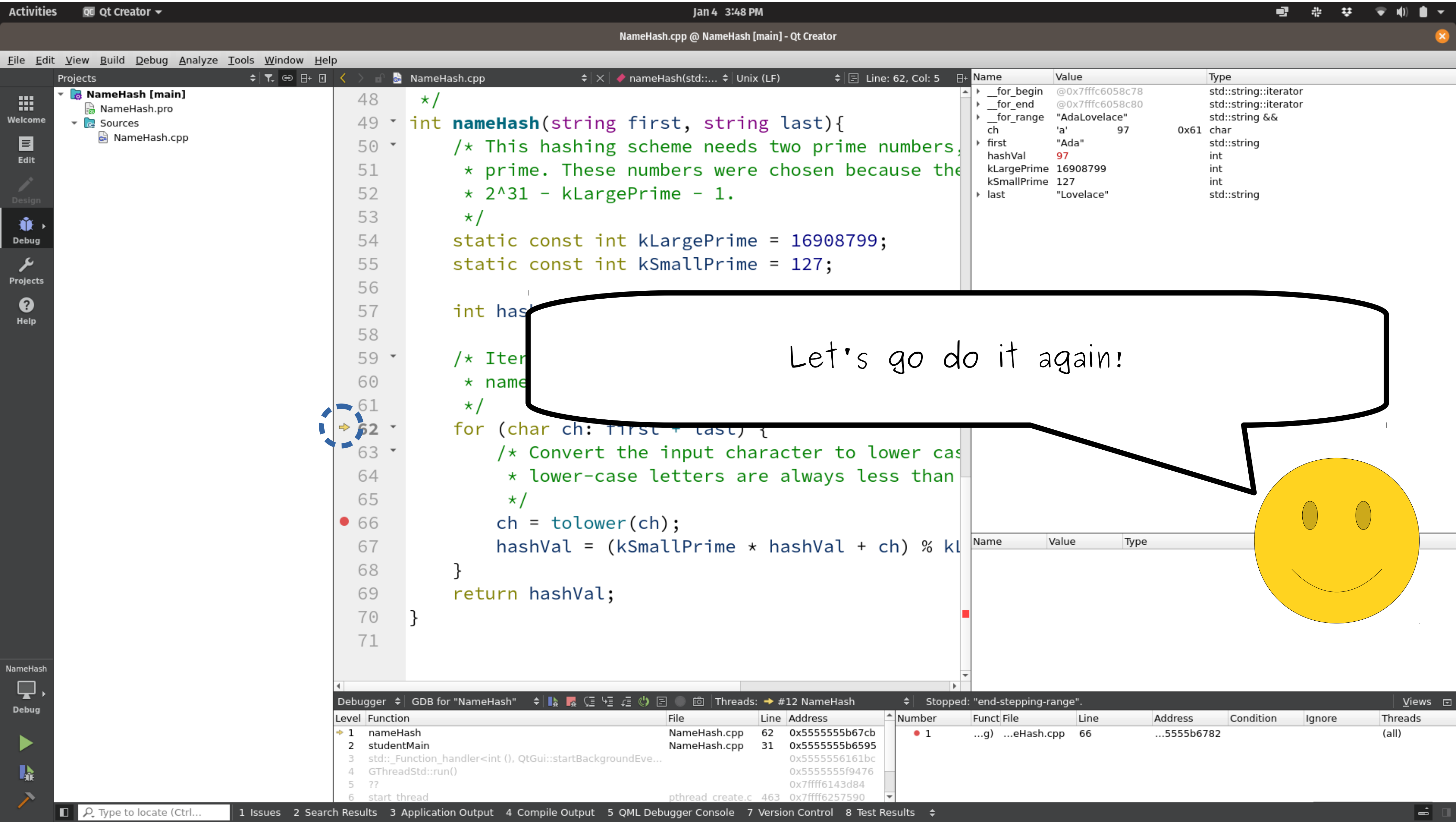
Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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



```
   */
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 has

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

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97 0x61	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

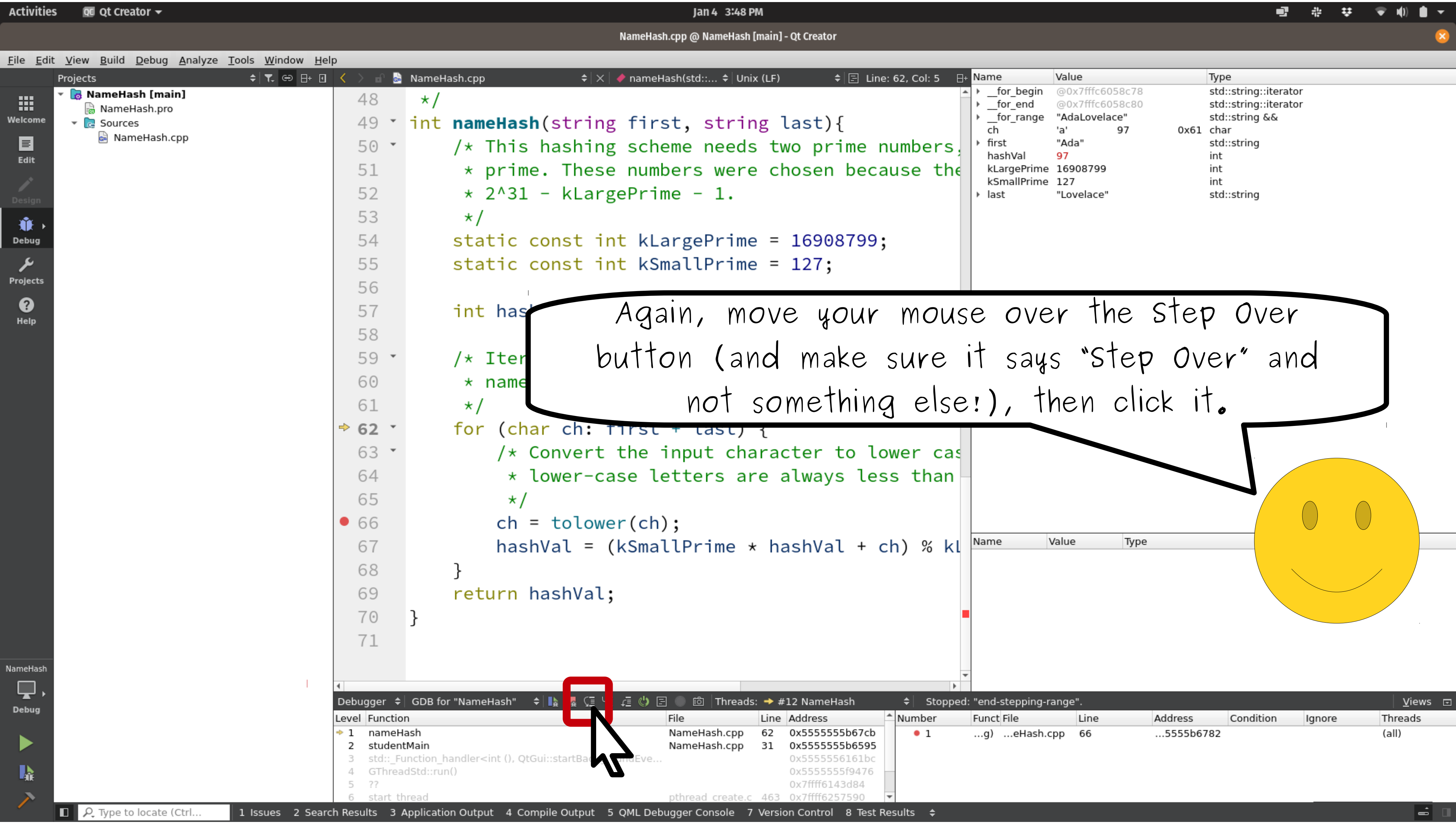
Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

Let's go do it again!



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

```
 */
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 has

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

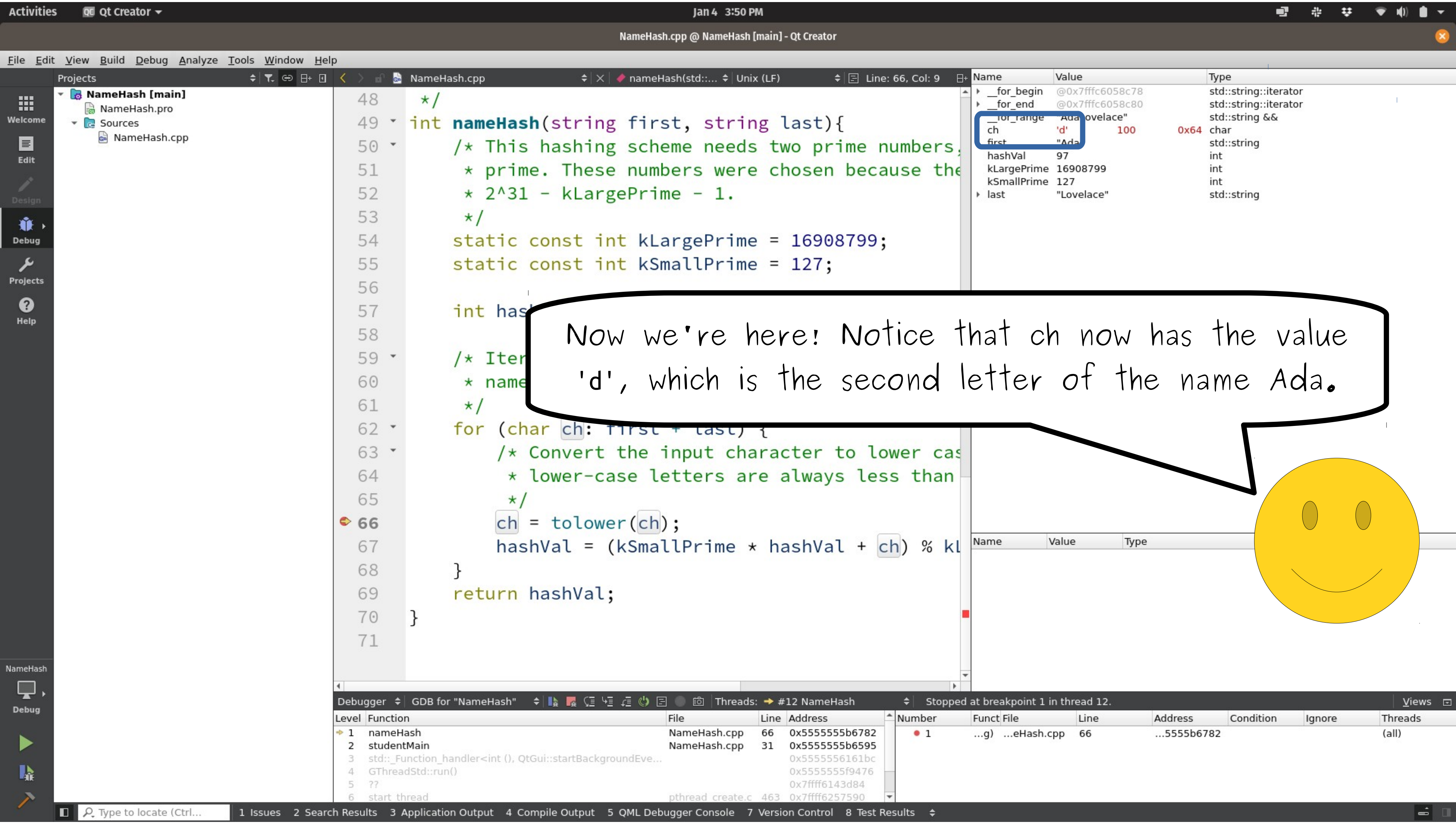
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'a'	97	0x61	char
first	"Ada"			std::string
hashVal	97			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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::startBa...ndEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



```
  */
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 has

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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			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

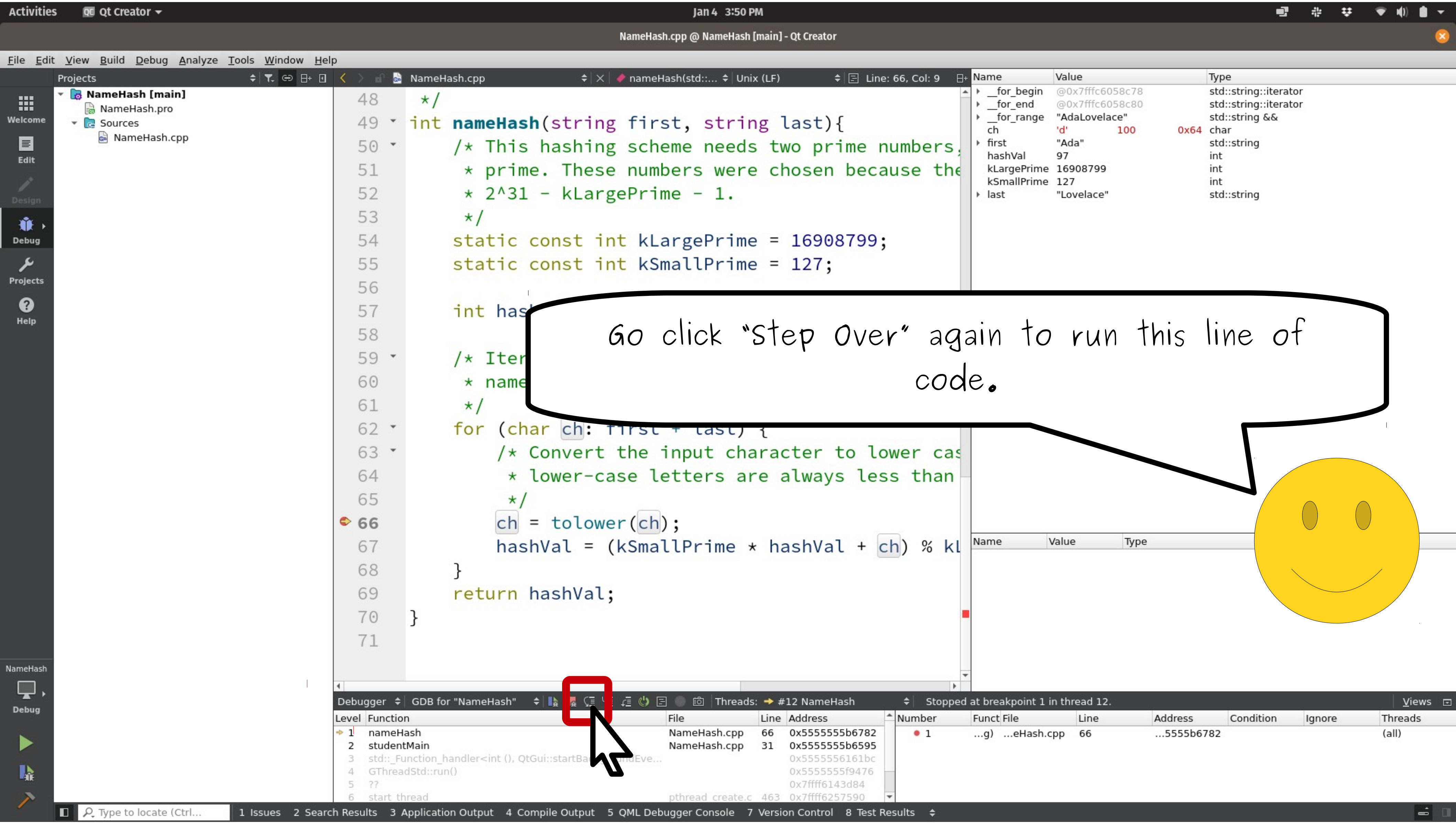
Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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

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



```
  */
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 has

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

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	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

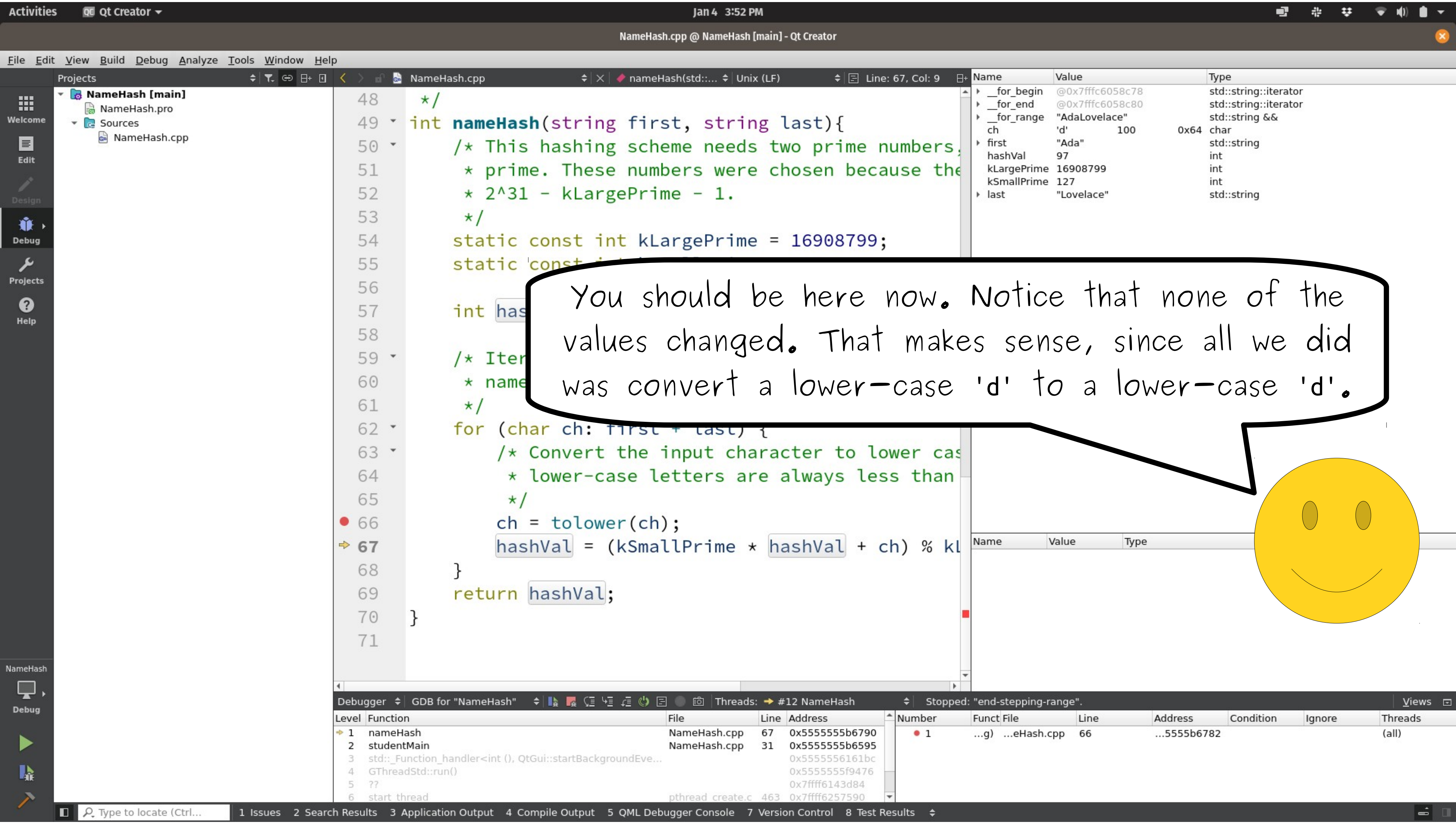
Name	Value	Type

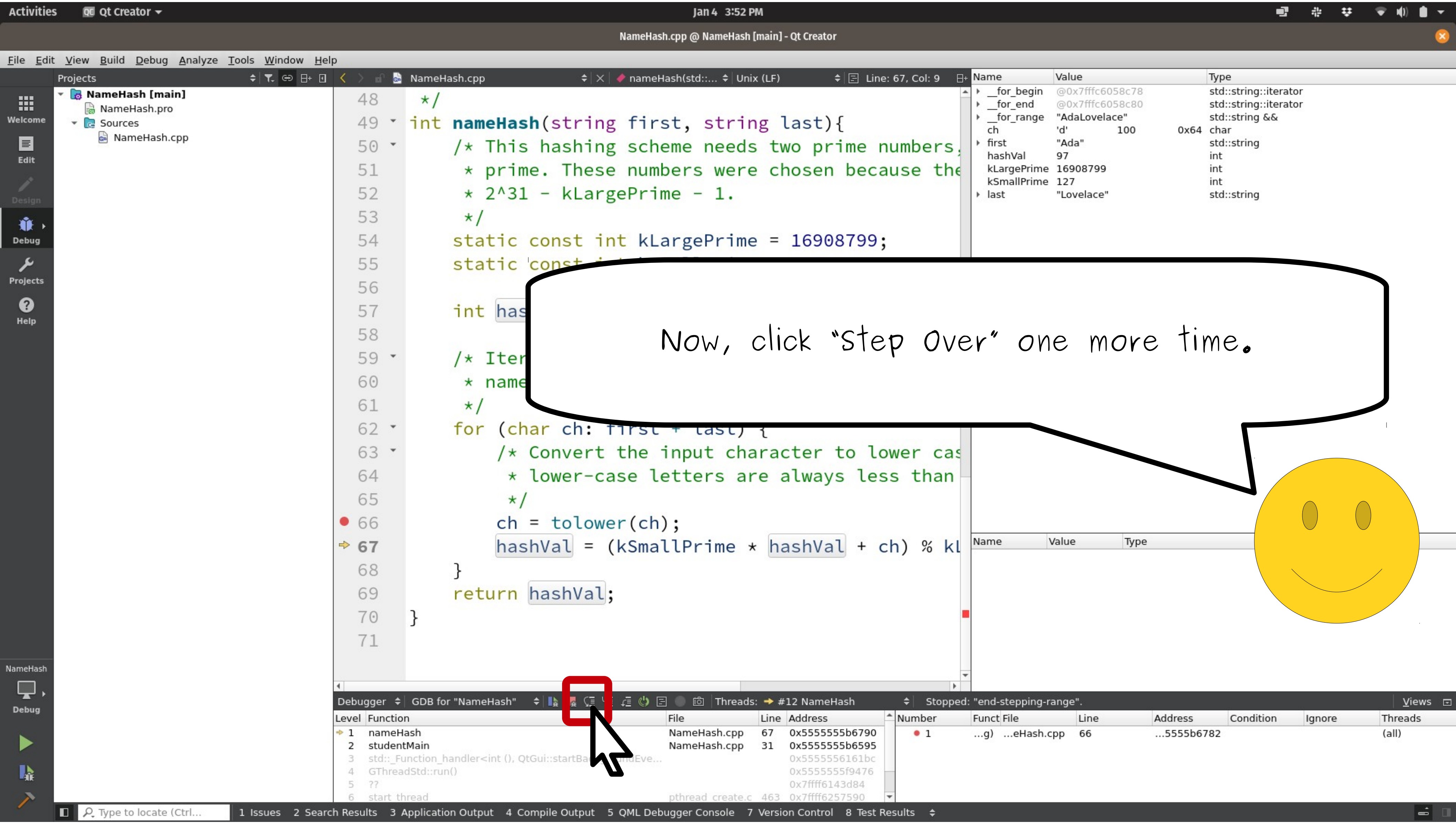
Stopped at breakpoint 1 in thread 12.

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

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

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





Now, click "Step Over" one more time.

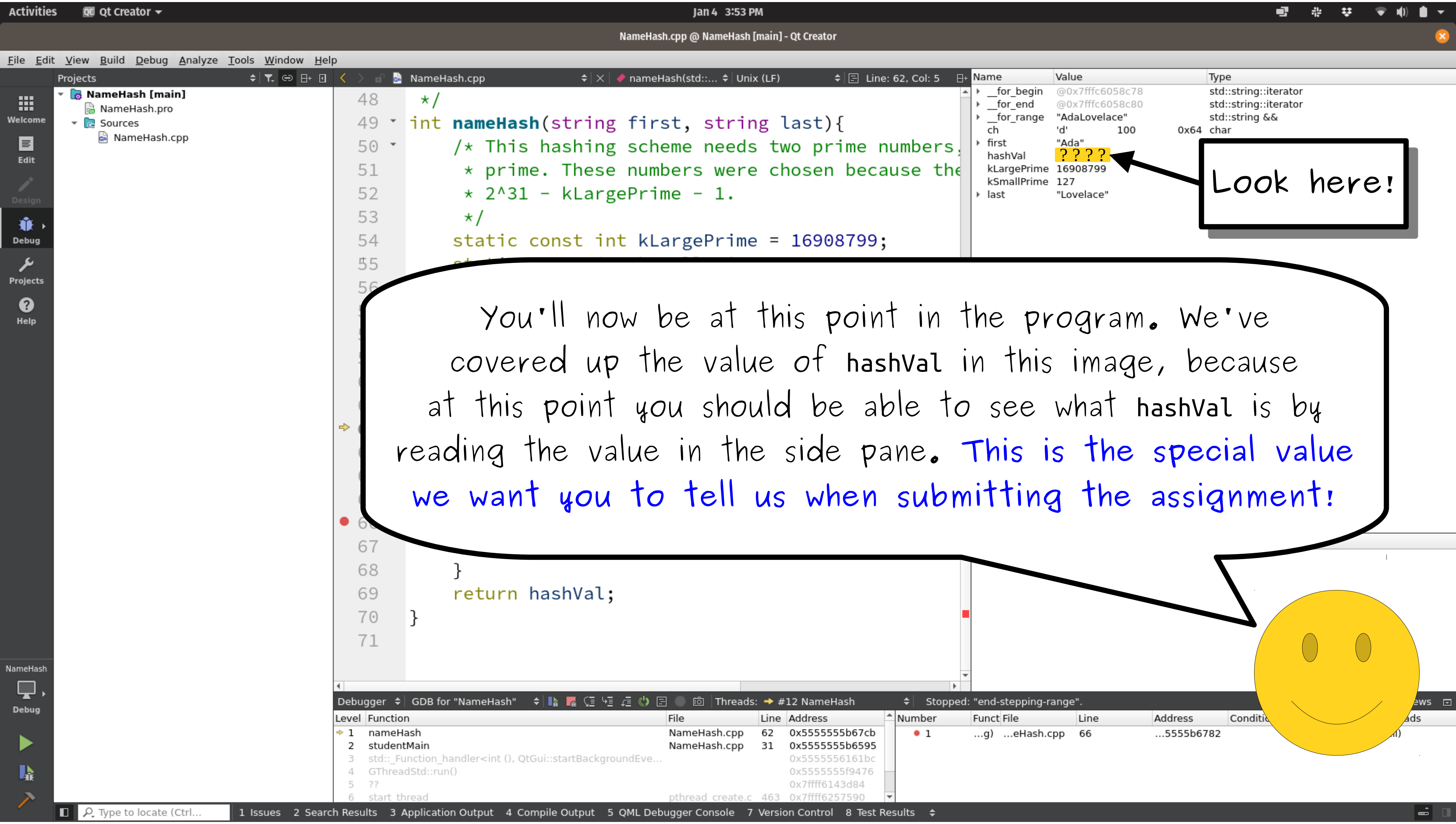
```
  */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const

    int has

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

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			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

Name	Value	Type

Level	Function	File	Line	Address
1	nameHash	NameHash.cpp	67	0x5555555b6790
2	studentMain	NameHash.cpp	31	0x5555555b6595
3	std::_Function_handler<int (), QtGui::startBa... ...ndEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



NameHash.cpp

```
  */
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;
```

```
    }
    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'd'	100	0x64	char
first	"Ada"			
hashVal	? ? ? ?			
kLargePrime	16908799			
kSmallPrime	127			
last	"Lovelace"			

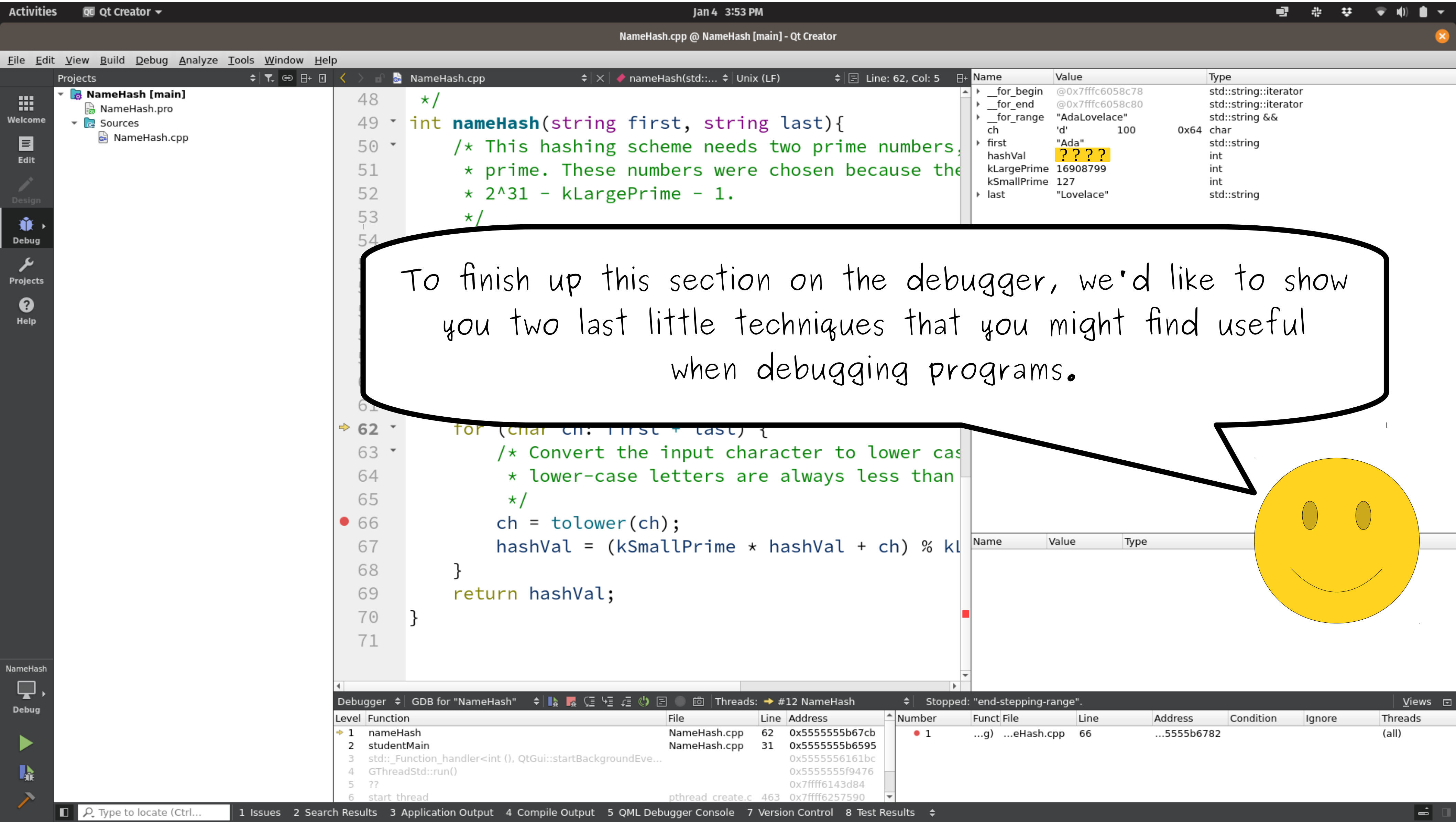
Look here!

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!**

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Number	Funct	File	Line	Address	Conditi
1	...g)	...eHash.cpp	66	...5555b6782	



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

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

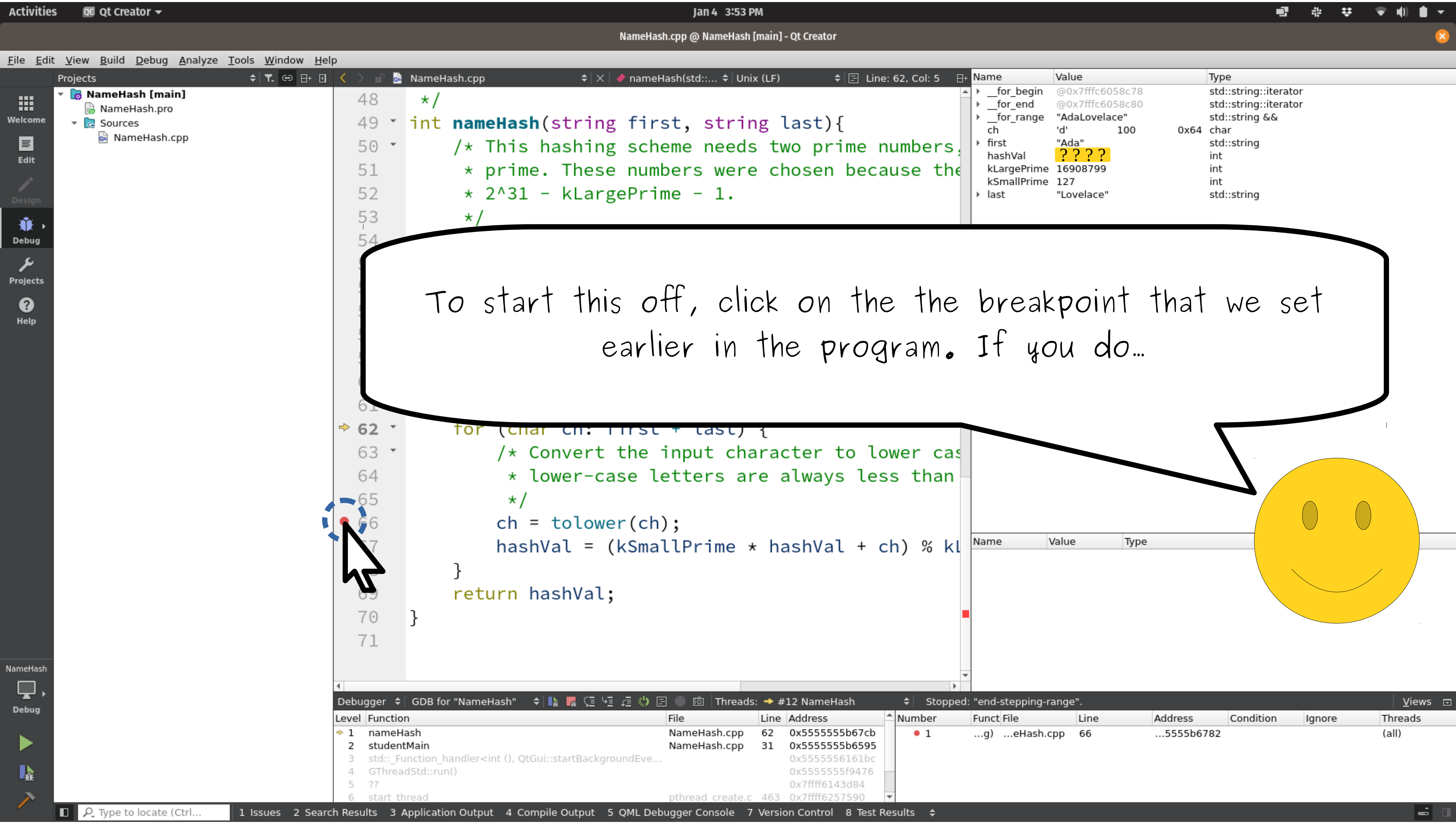
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'd'	100	0x64	char
first	"Ada"			std::string
hashVal	? ? ? ?			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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

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

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



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

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

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

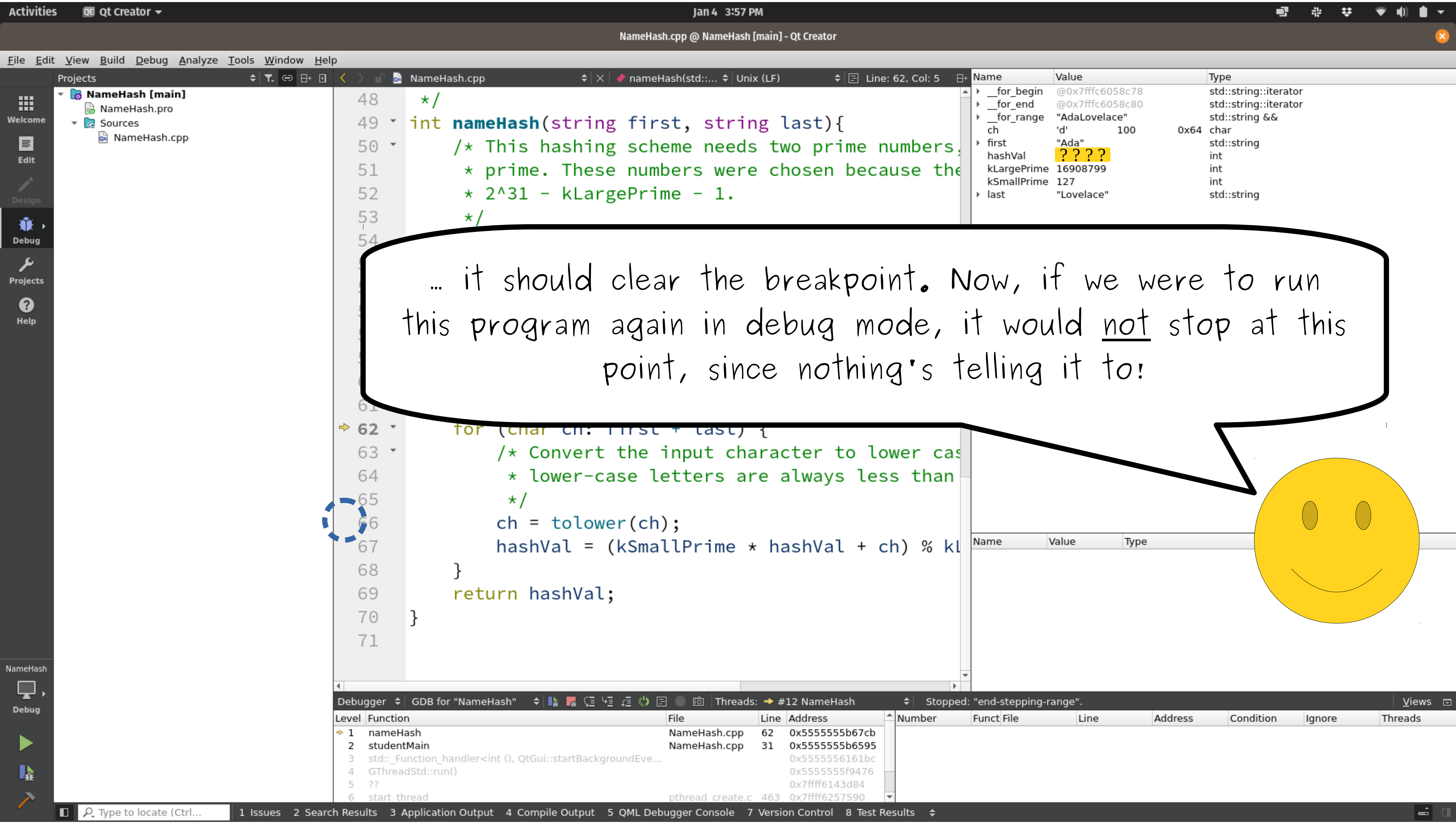
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'd'	100	0x64	char
first	"Ada"			std::string
hashVal	? ? ? ?			int
kLargePrime	16908799			int
kSmallPrime	127			int
last	"Lovelace"			std::string

Name	Value	Type

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

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



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

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

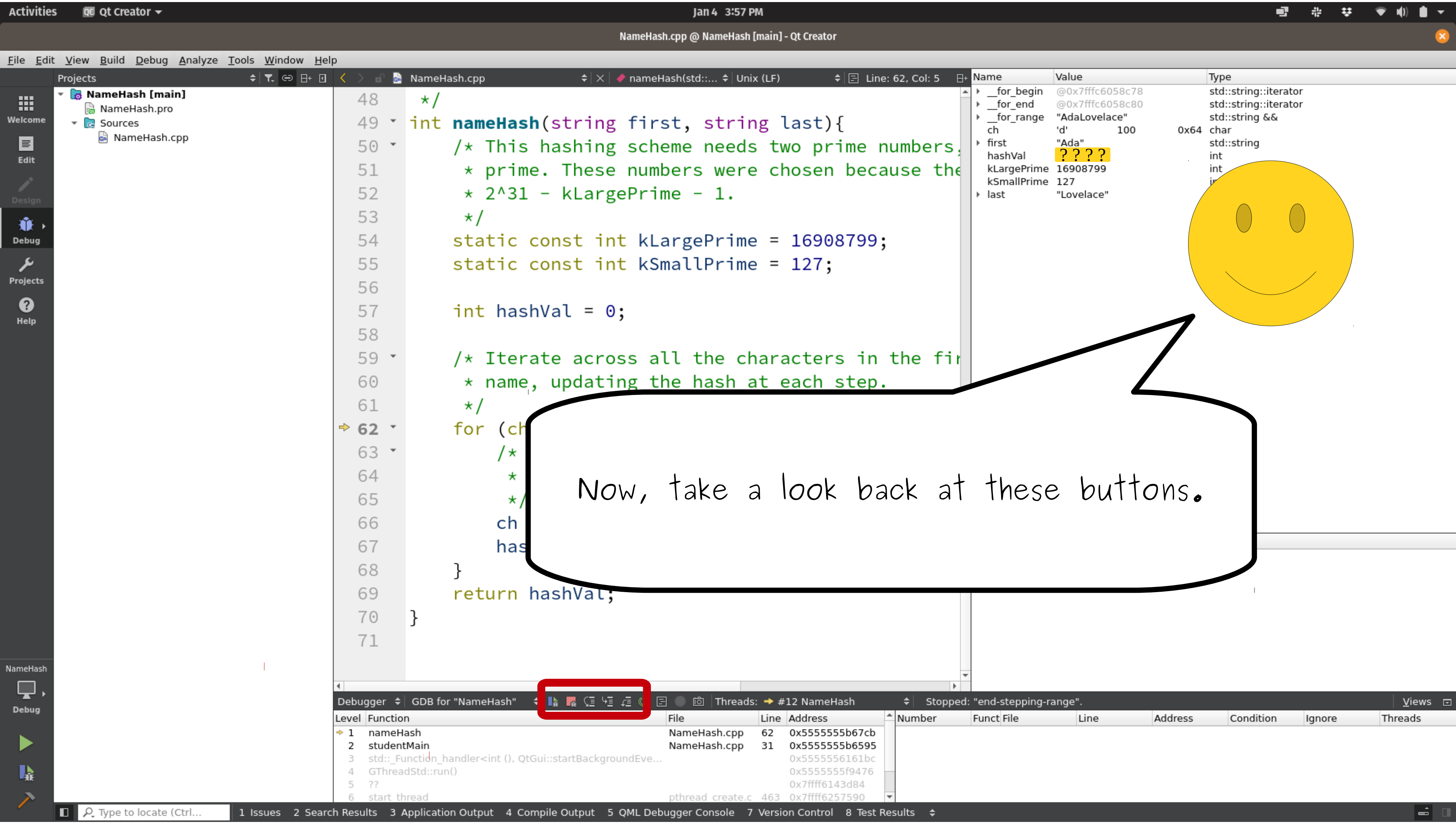
Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'd' 100 0x64	char
first	"Ada"	std::string
hashVal	????	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	"Lovelace"	std::string

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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



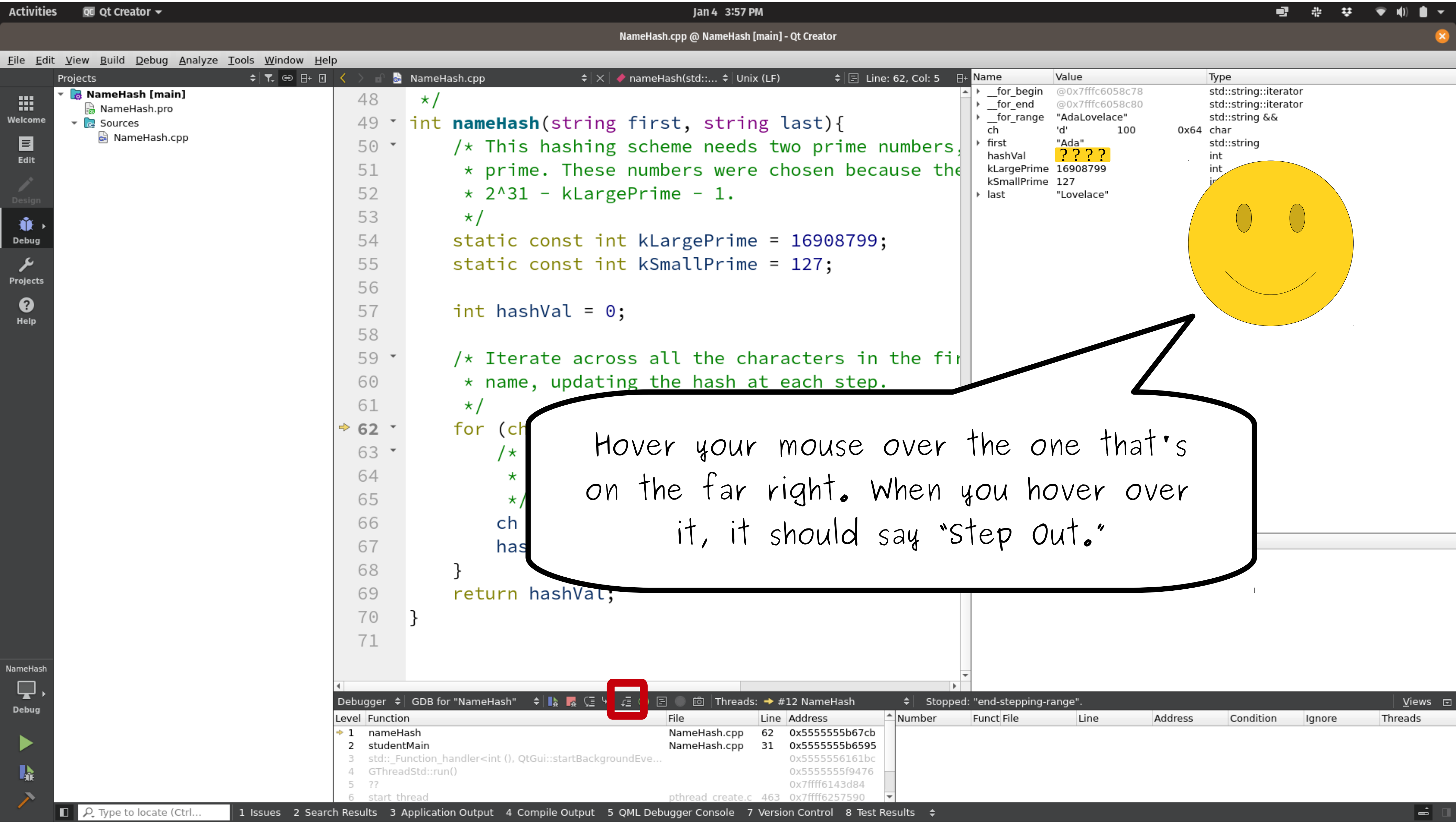
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fir
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

Name	Value	Type
__for_begin	@0x7fffc6058c78	std::string::iterator
__for_end	@0x7fffc6058c80	std::string::iterator
__for_range	"AdaLovelace"	std::string &&
ch	'd' 100 0x64	char
first	"Ada"	std::string
hashVal	? ? ? ?	int
kLargePrime	16908799	int
kSmallPrime	127	i
last	"Lovelace"	

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

Number	Funct	File	Line	Address	Condition	Ignore	Threads

Now, take a look back at these buttons.



```
 */
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 fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

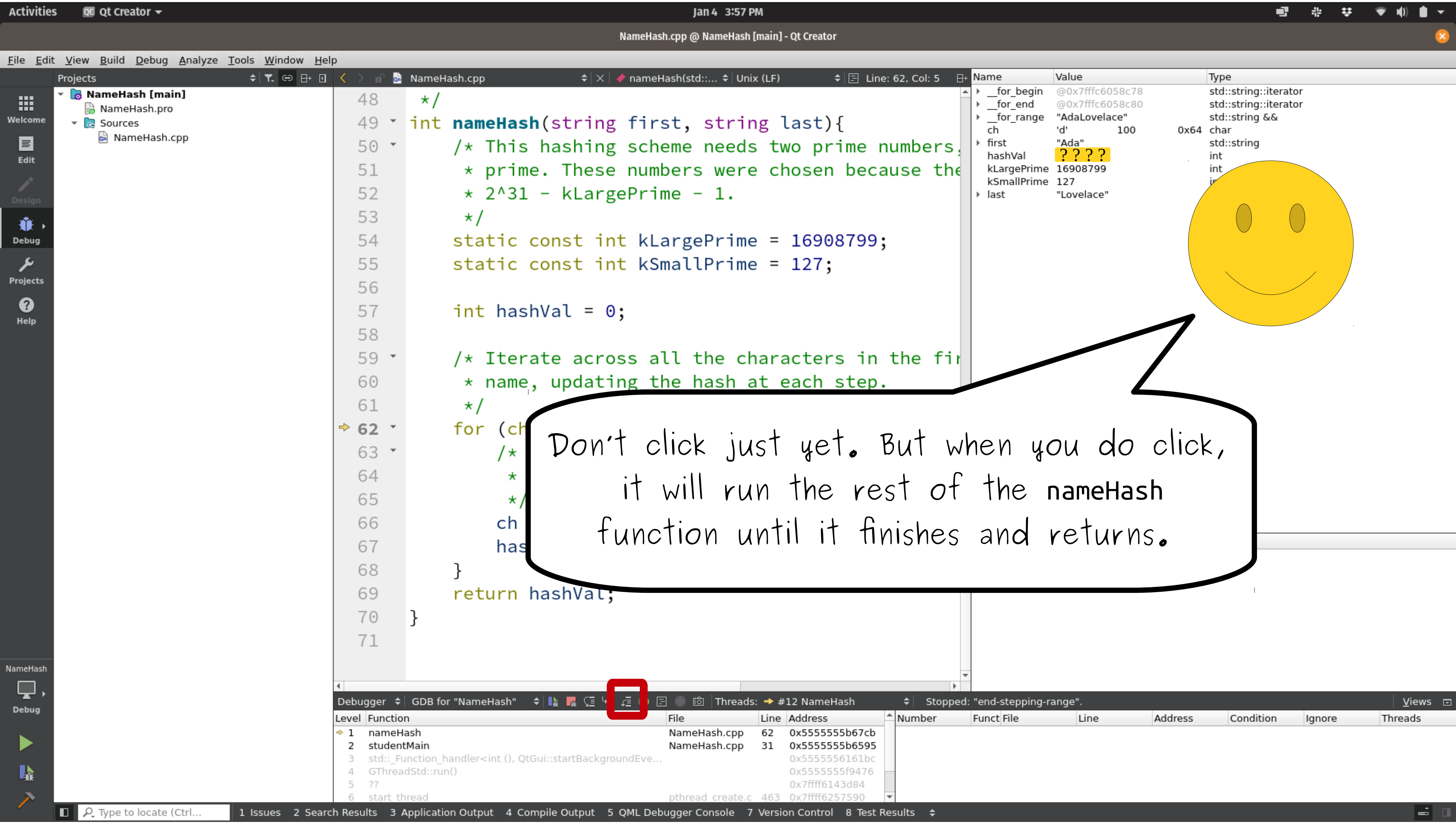
Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'd'	100	0x64	char
first	"Ada"			std::string
hashVal	? ? ? ?			int
kLargePrime	16908799			int
kSmallPrime	127			i
last	"Lovelace"			

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::startBackgroundEve...			0x5555556161bc
4	GThreadStd::run()			0x5555555f9476
5	??			0x7ffff6143d84
6	start_thread	pthread_create.c	463	0x7ffff6257590

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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



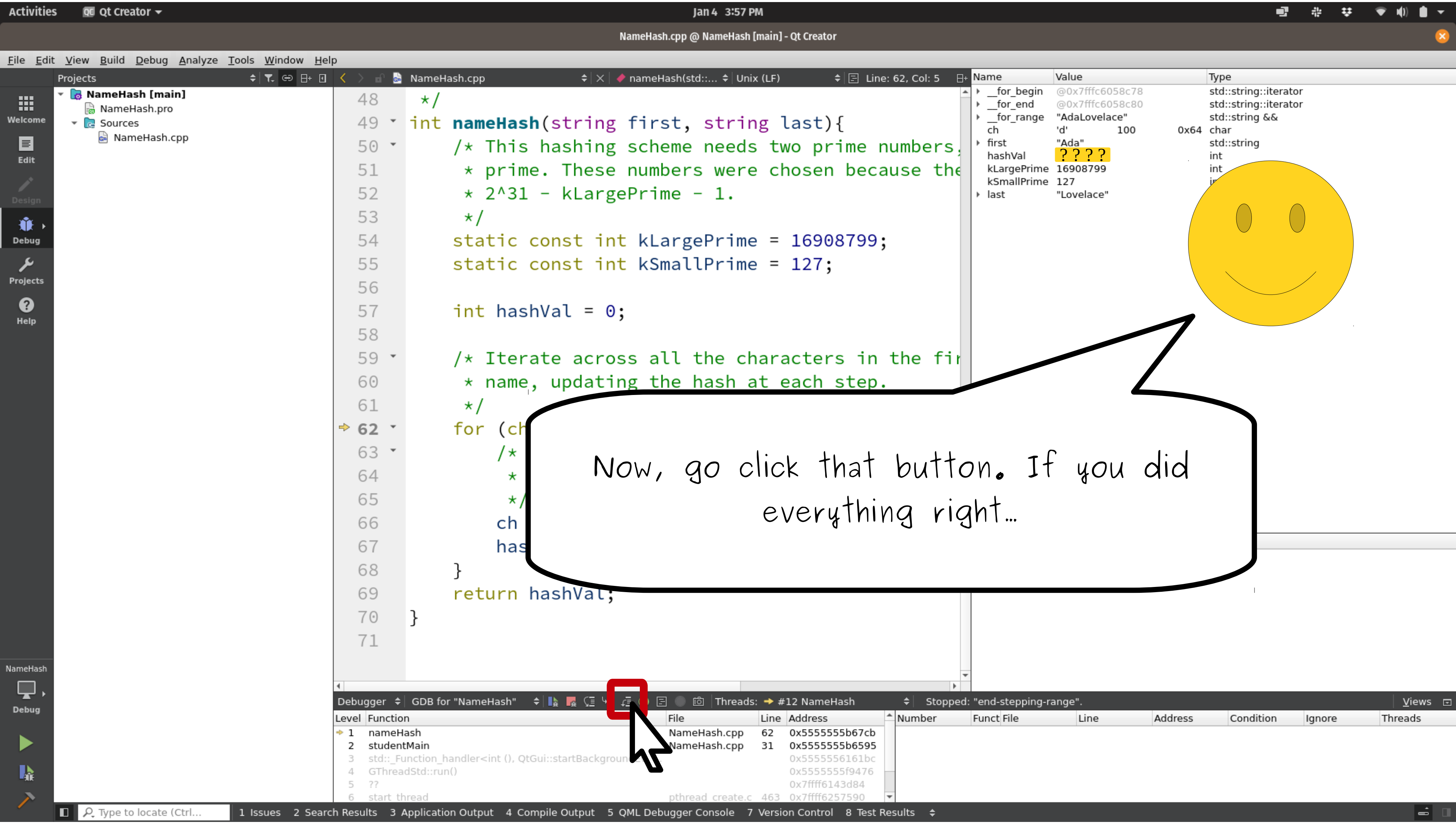
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'd'	100	0x64	char
first	"Ada"			std::string
hashVal	? ? ? ?			int
kLargePrime	16908799			int
kSmallPrime	127			i
last	"Lovelace"			

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

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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



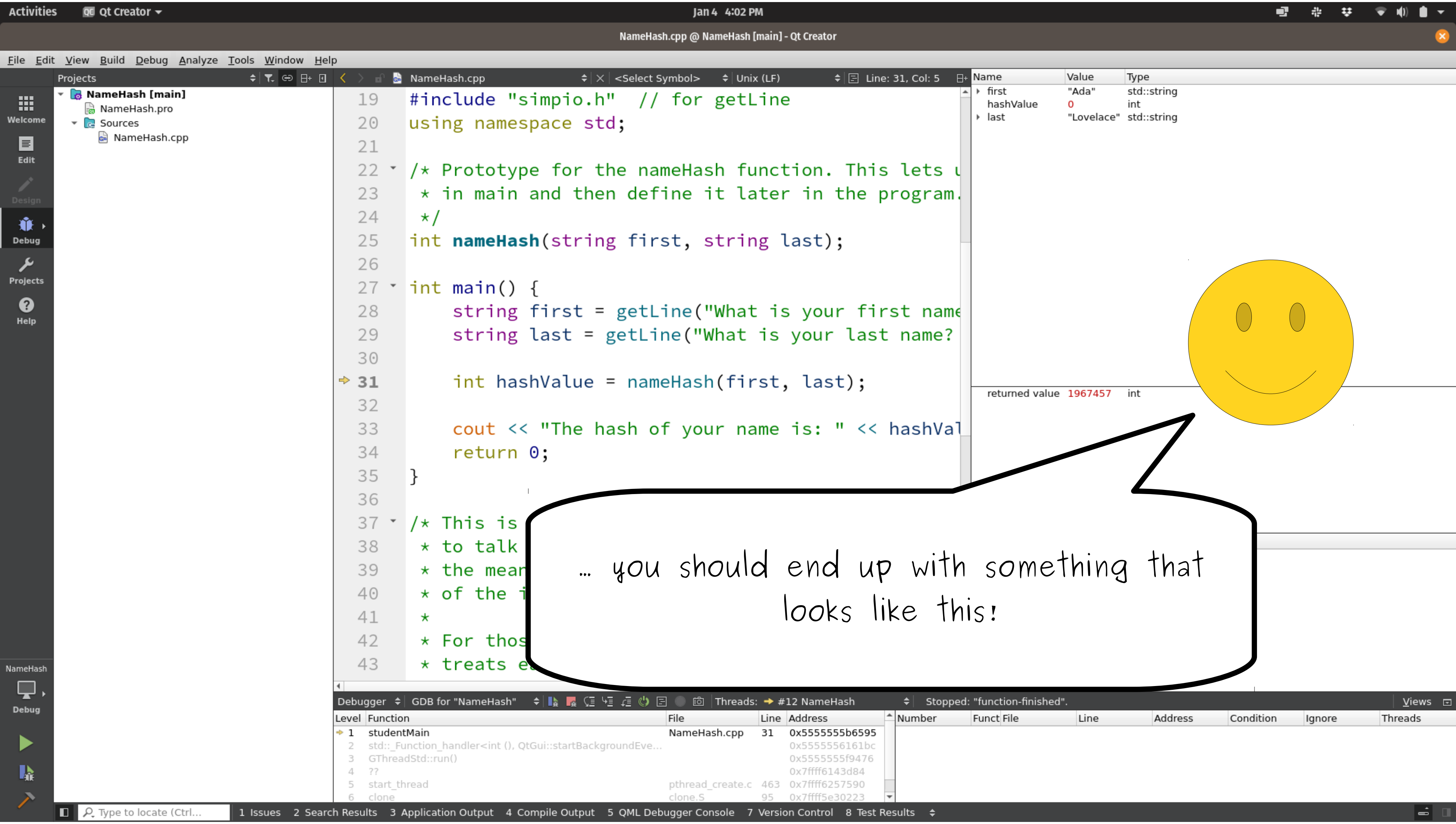
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers,
     * prime. These numbers were chosen because the
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the fi
     * name, updating the hash at each step.
     */
    for (ch
        /*
         *
         */
        ch
        has
    }
    return hashVal;
}
```

Name	Value			Type
__for_begin	@0x7fffc6058c78			std::string::iterator
__for_end	@0x7fffc6058c80			std::string::iterator
__for_range	"AdaLovelace"			std::string &&
ch	'd'	100	0x64	char
first	"Ada"			std::string
hashVal	? ? ? ?			int
kLargePrime	16908799			int
kSmallPrime	127			in
last	"Lovelace"			

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

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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



```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is
 * to talk
 * the mear
 * of the i
 *
 * For thos
 * treats e
```

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

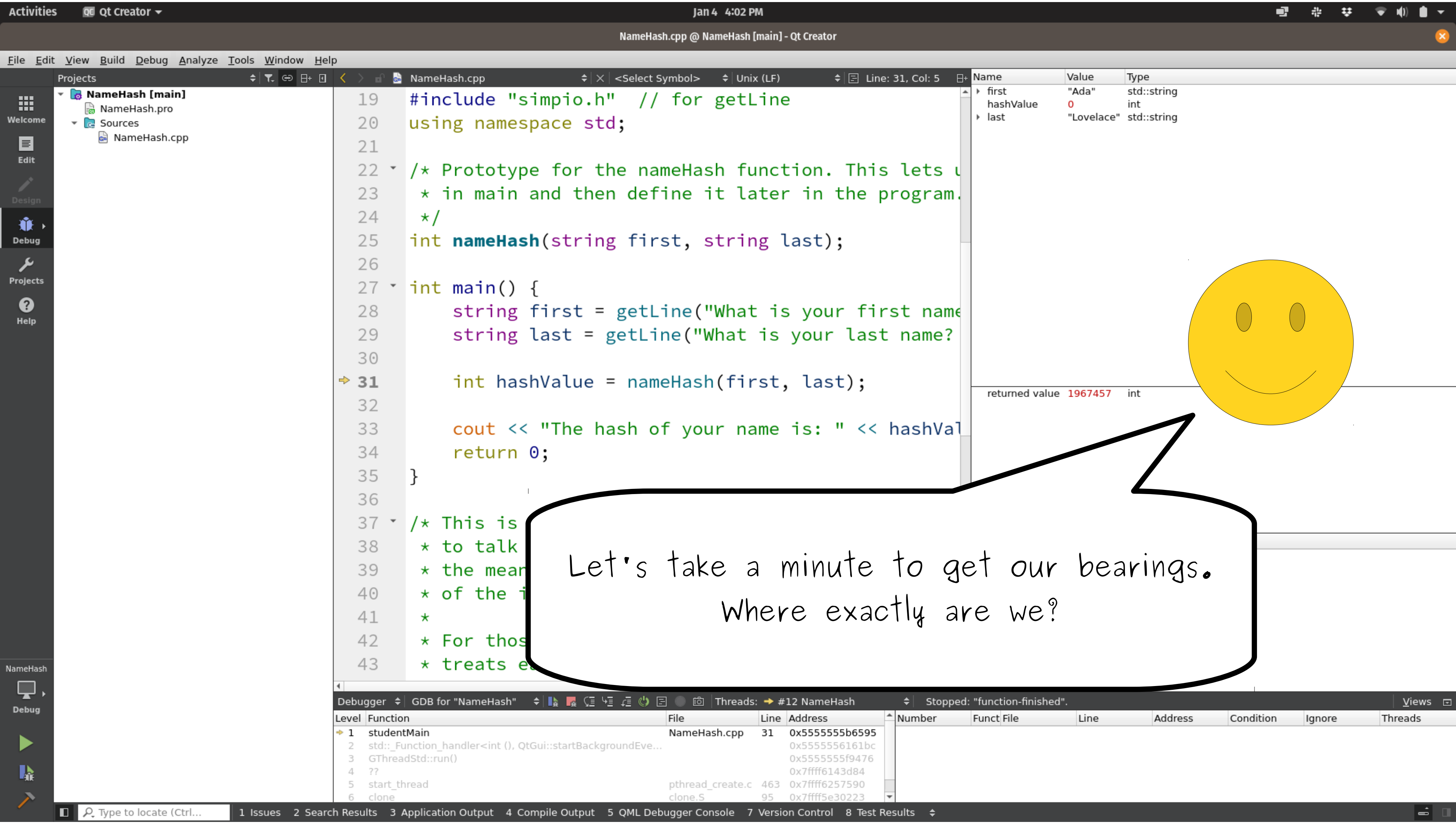
returned value	1967457	int

Threads: #12 NameHash

Stopped: "function-finished".

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

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]
NameHash.pro
Sources
NameHash.cpp
<Select Symbol>
Unix (LF)
Line: 31, Col: 5
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Debugger
GDB for "NameHash"
Views
Number Funct File Line Address Condition Ignore Threads
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 end up with something that looks like this!


Activities
Qt Creator
Jan 4 4:02 PM
File Edit View Build Debug Analyze Tools Window Help
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Let's take a minute to get our bearings. Where exactly are we?

NameHash.cpp @ NameHash [main] - Qt Creator

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

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

```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is
 * to talk
 * the mean
 * of the i
 *
 * For thos
 * treats e
```

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

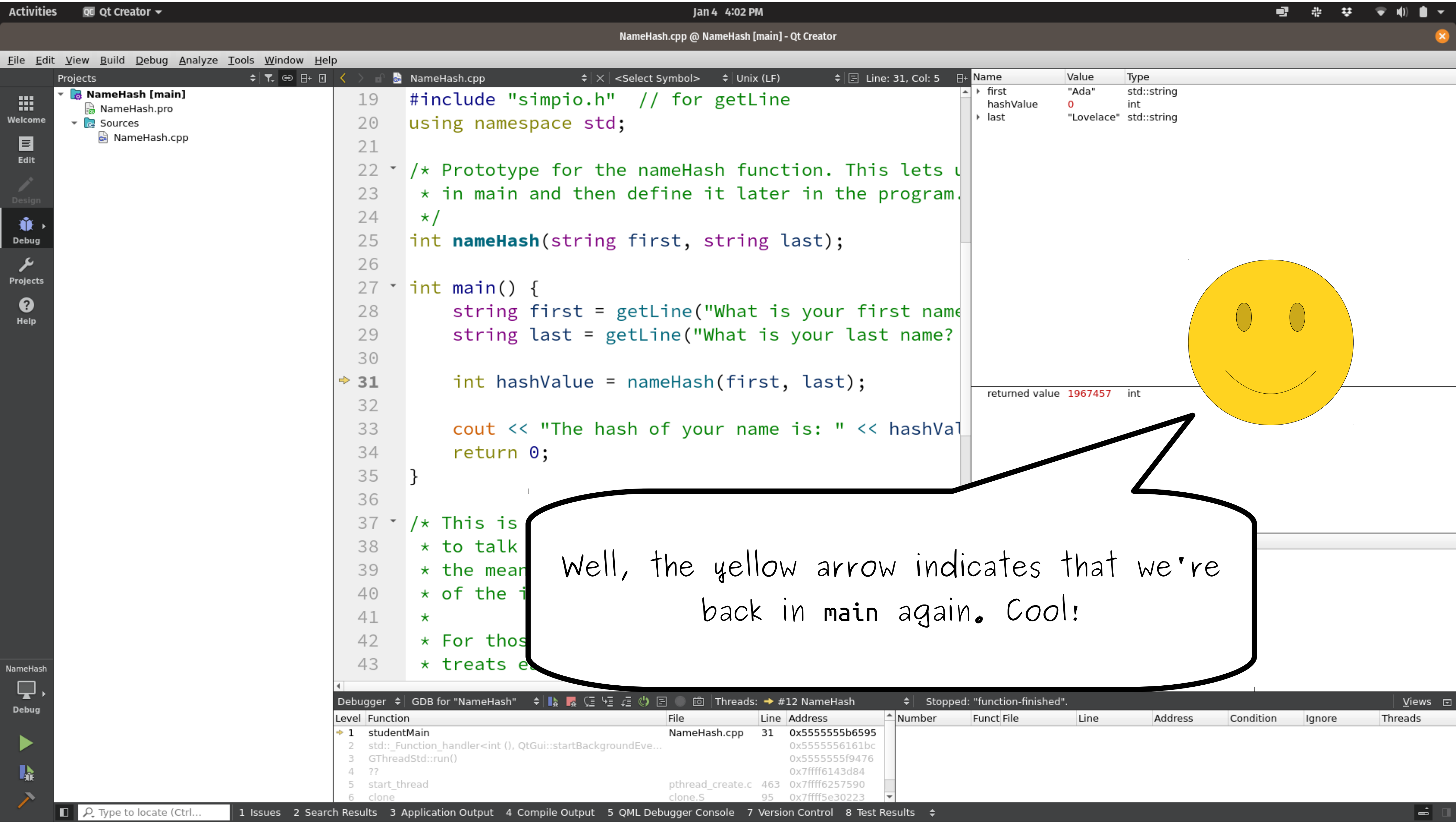
returned value 1967457 int

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 (), QtGui::startBackgroundEve...			0x5555556161bc
3	GThreadStd::run()			0x5555555f9476
4	??			0x7ffff6143d84
5	start_thread	pthread_create.c	463	0x7ffff6257590
6	clone	clone.S	95	0x7ffff5e30223

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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



```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is
 * to talk
 * the mear
 * of the i
 *
 * For thos
 * treats e
```

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

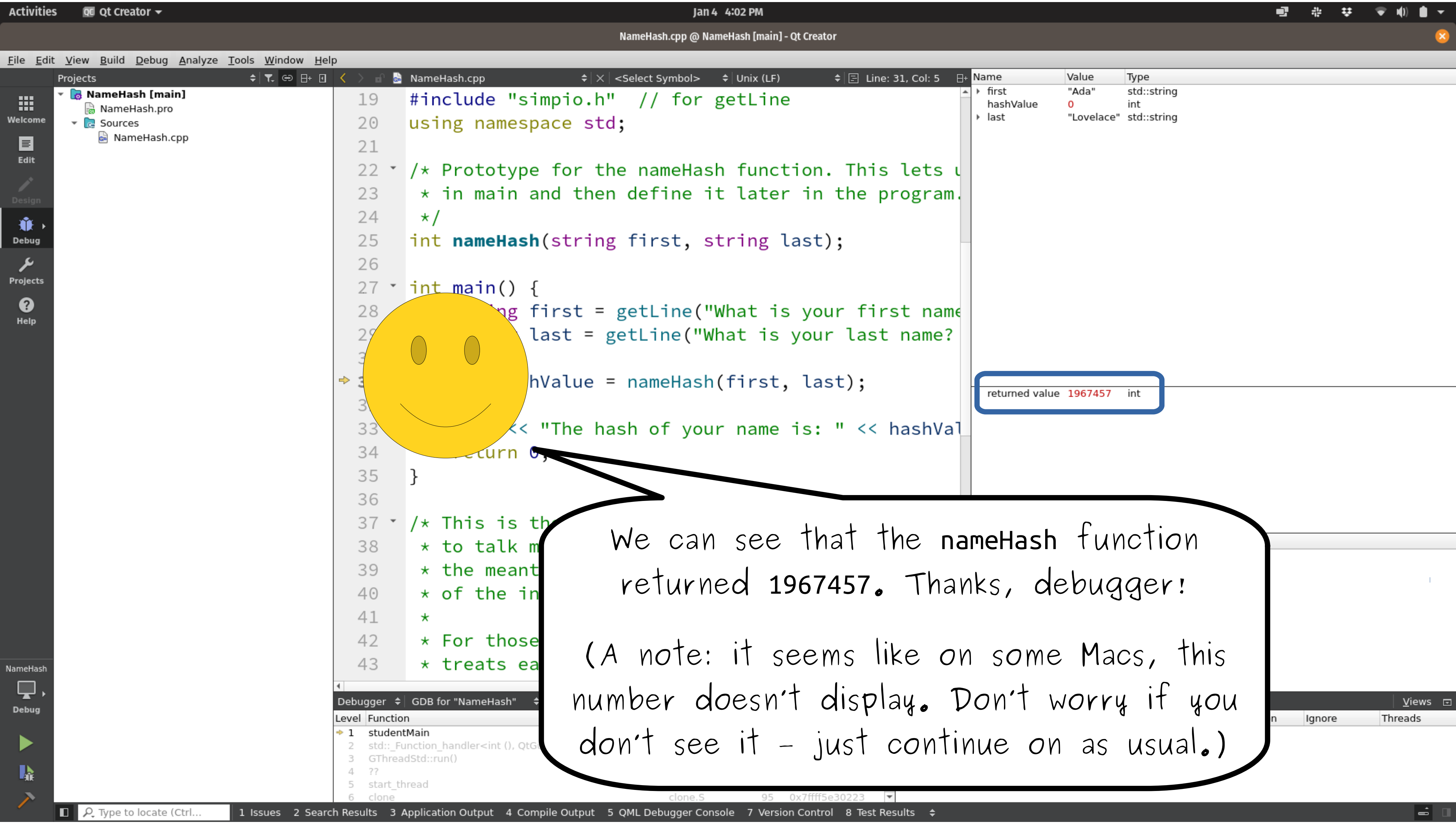
returned value	1967457	int

Stopped: "function-finished".

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

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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



NameHash.cpp @ NameHash [main] - Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Welcome
Edit
Design
Debug
Projects
Help

```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * in main and then define it later in the program.
 */
int nameHash(string first, string last);

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

           hValue = nameHash(first, last);

         << "The hash of your name is: " << hashVal
    return 0;
}

/* This is th
 * to talk m
 * the meant
 * of the in
 *
 * For those
 * treats ea
```

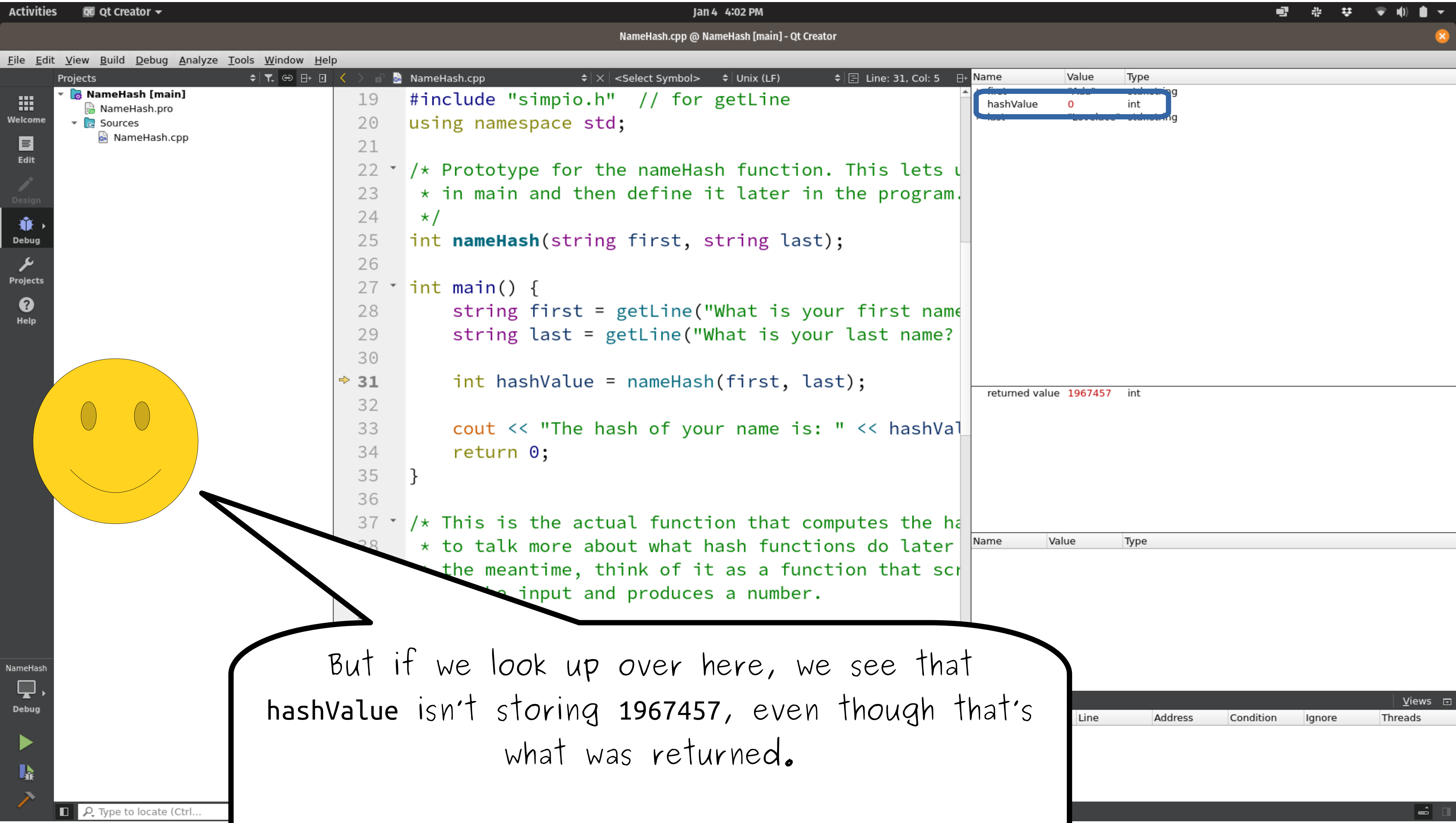
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

```
returned value  1967457   int
```

Level	Function
1	studentMain
2	std::_Function_handler<int (), QtG
3	GThreadStd::run()
4	??
5	start_thread
6	clone

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.)



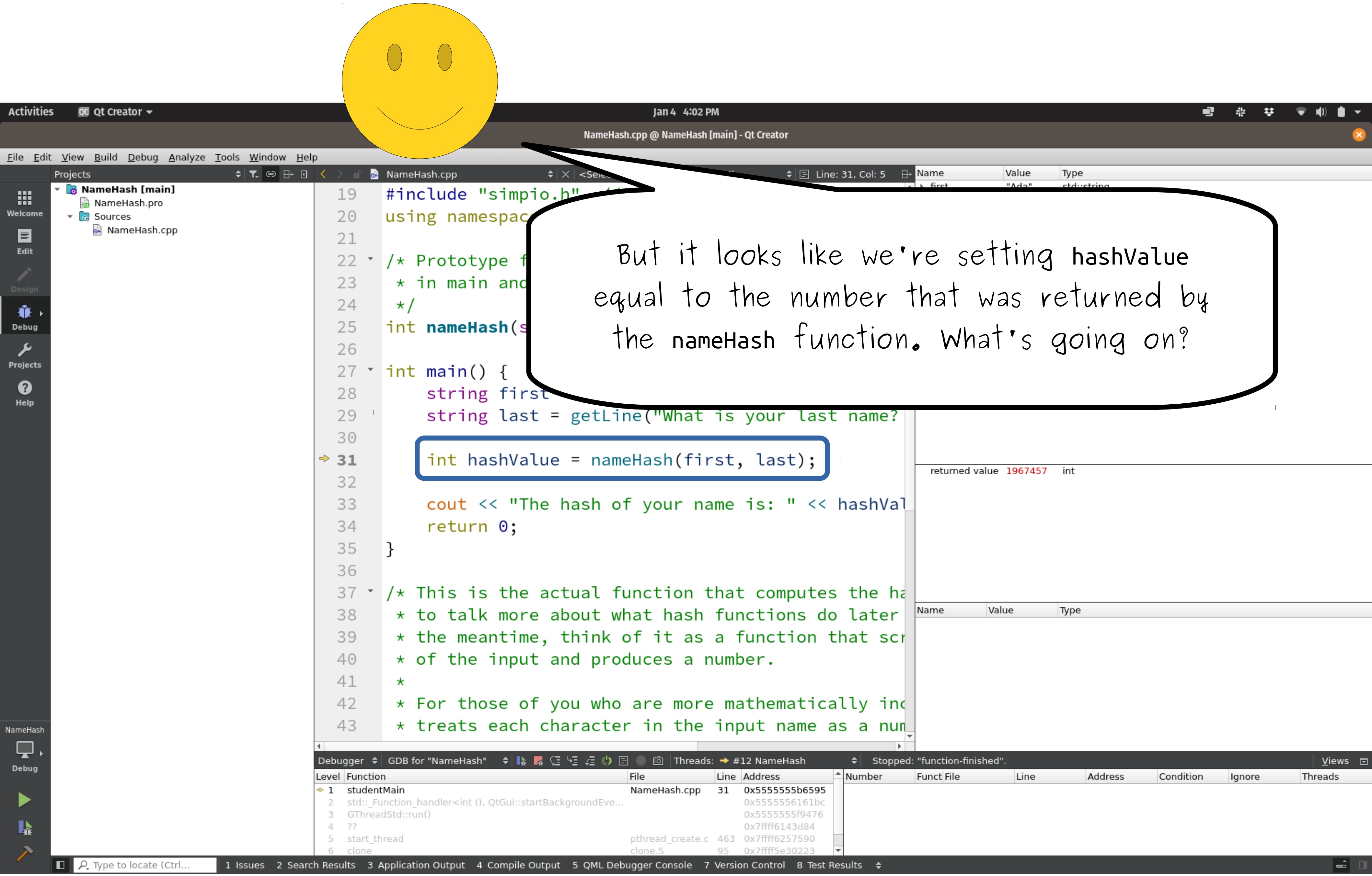
```
#include "simpio.h"  // for getLine
using namespace std;

/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * input and produces a number.
```

Name	Value	Type
hashValue	0	int

returned value	1967457	int

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.)


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]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Line: 31, Col: 5
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results
Type to locate (Ctrl...
Views

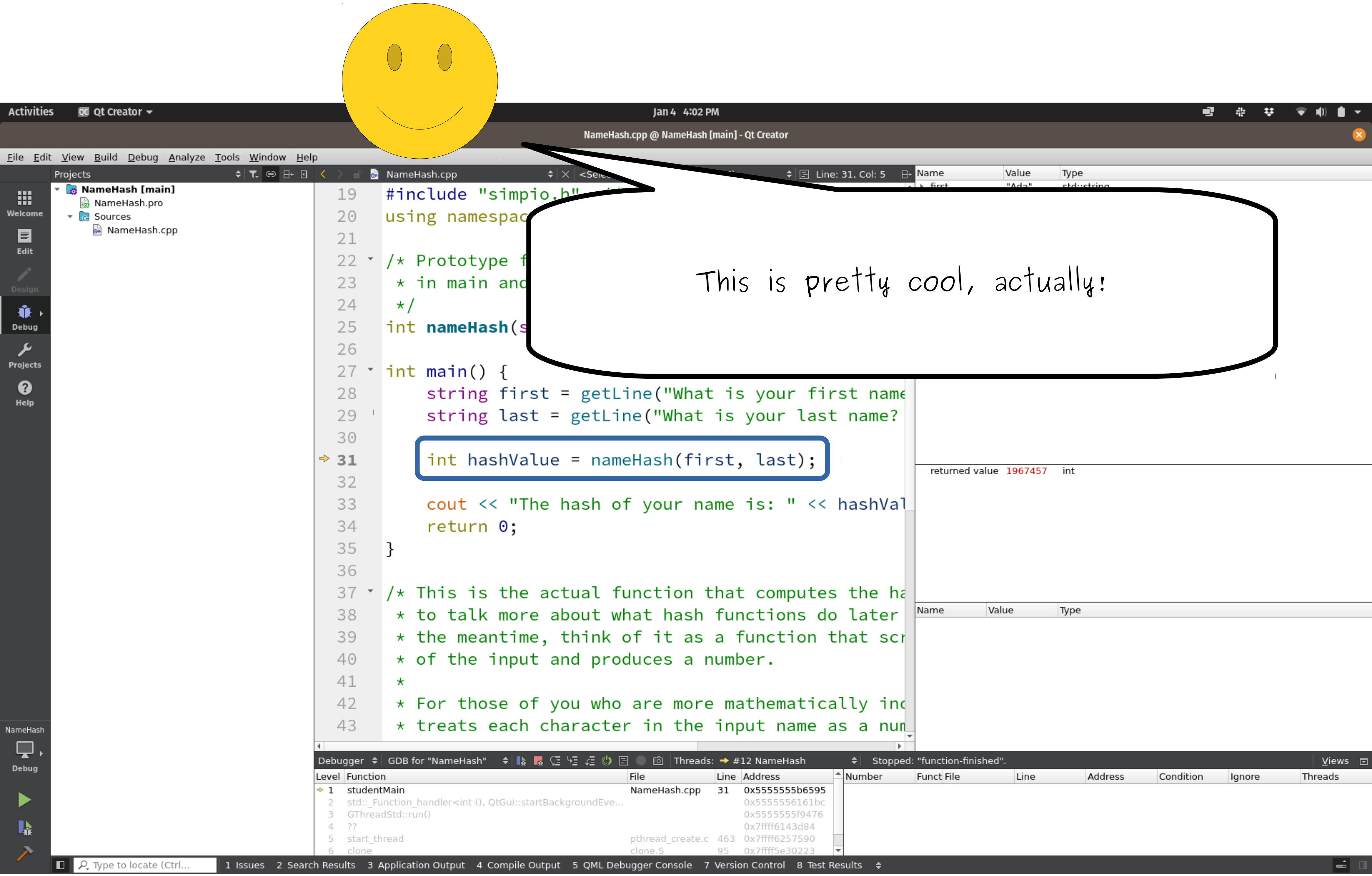
But it looks like we're setting `hashValue` equal to the number that was returned by the `nameHash` function. What's going on?

```
#include "simpio.h"
using namespac

/* Prototype f
 * in main and
 */
int nameHash(s

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

    int hashValue = nameHash(first, last);

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

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a nun
```

```
Name            Value      Type
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 (), QtGui::startBackgroundEve...                   0x5555556161bc
3     GThreadStd::run()                                                             0x5555555f9476
4     ??                                                                            0x7ffff6143d84
5     start_thread                                           pthread_create.c 463   0x7ffff6257590
6     clone                                                  clone.S          95    0x7ffff5e30223
```

```
Number  Funct File  Line  Address  Condition  Ignore  Threads
```



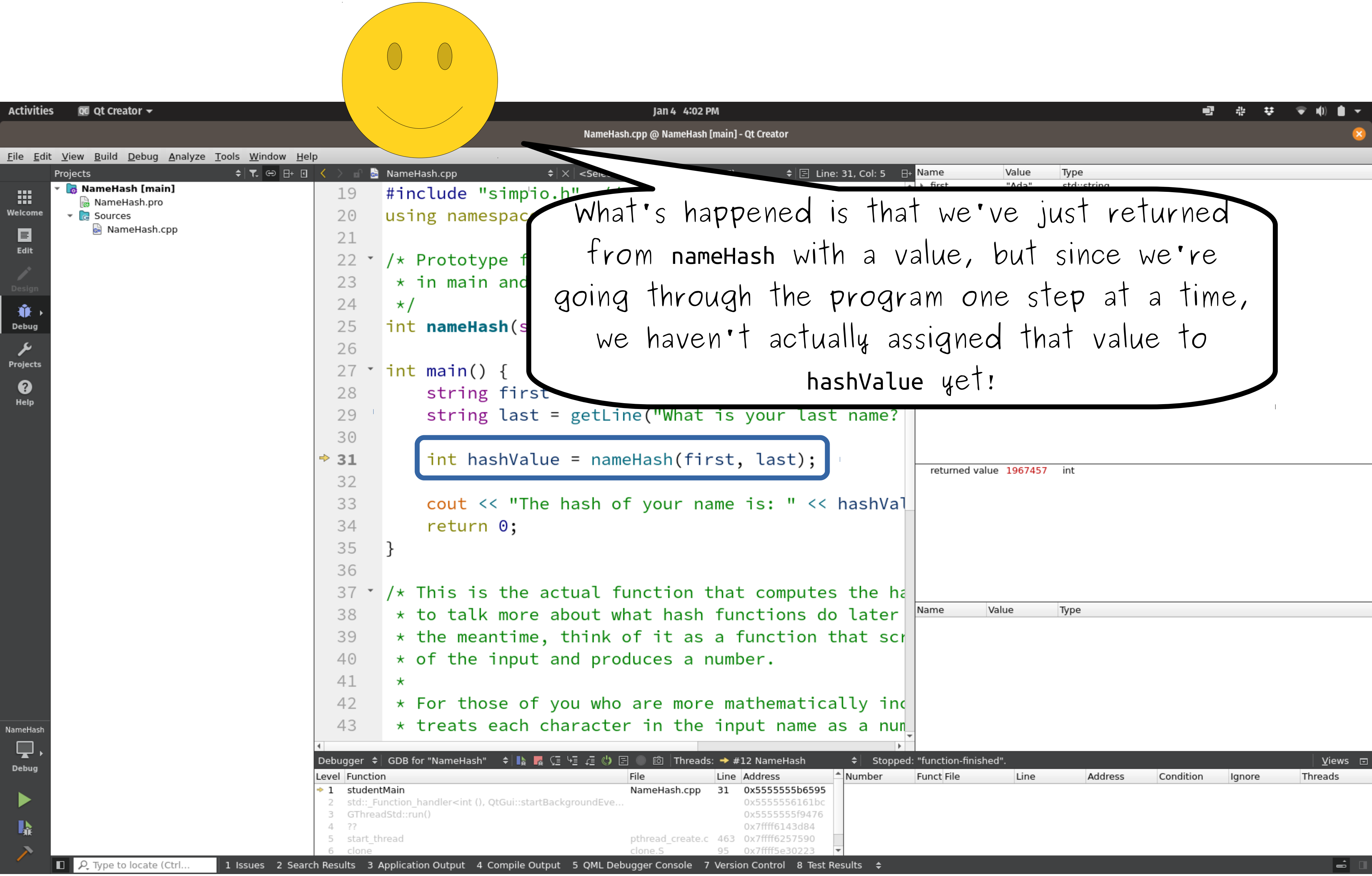
```
#include "simpio.h"
using namespac

/* Prototype f
 * in main and
 */
int nameHash(s

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: " << hashVal
    return 0;
}

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a nun
```

Name	Value	Type
first	"Ada"	std::string

returned value	1967457	int

Stopped: "function-finished".

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



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!

Activities
Qt Creator
Jan 4 4:02 PM
NameHash.cpp @ NameHash [main] - Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Line: 31, Col: 5
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results
Type to locate (Ctrl...
Views

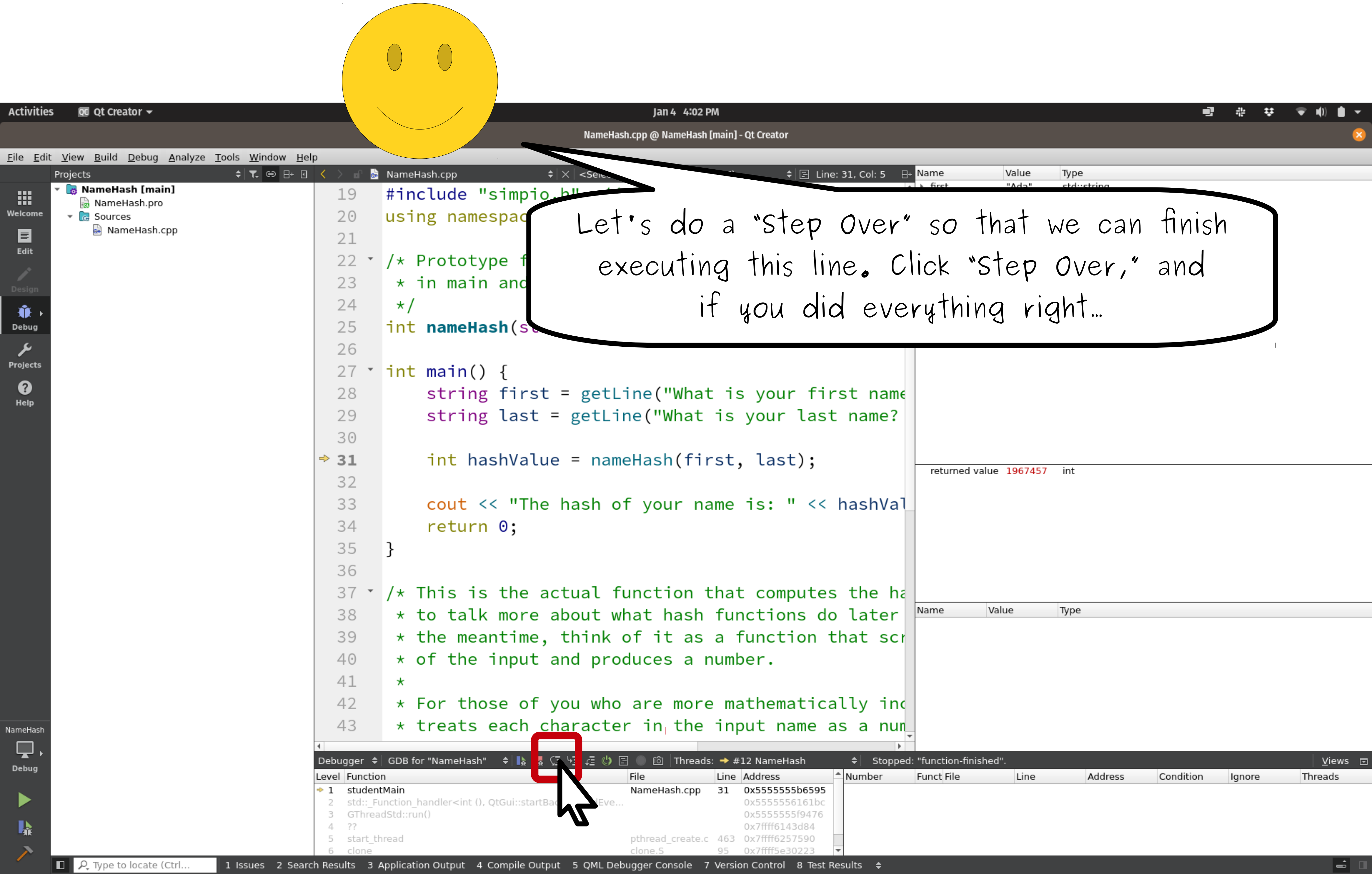
```
#include "simpio.h"
using namespac

/* Prototype f
 * in main and
 */
int nameHash(s

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

    int hashValue = nameHash(first, last);

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

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a num
```

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 (), QtGui::startBackgroundEve...			0x5555556161bc
3	GThreadStd::run()			0x5555555f9476
4	??			0x7ffff6143d84
5	start_thread	pthread_create.c	463	0x7ffff6257590
6	clone	clone.S	95	0x7ffff5e30223

Number	Funct	File	Line	Address	Condition	Ignore	Threads



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

```
#include "simpio.h"
using namespac

/* Prototype f
 * in main and
 */
int nameHash(st

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: " << hashVal
    return 0;
}

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a nun
```

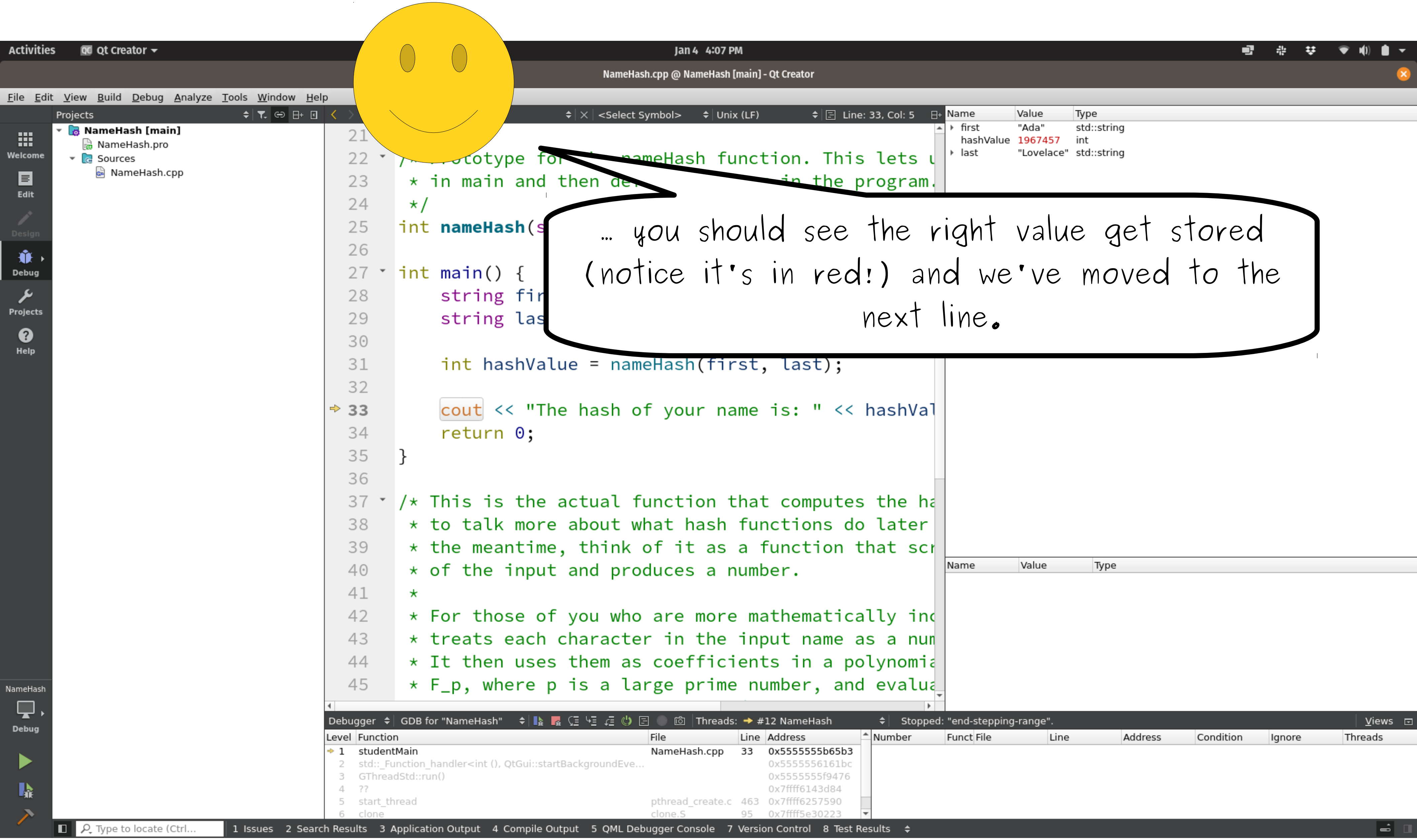
Name	Value	Type
first	"Ada"	std::string
returned value	1967457	int

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 (), QtGui::startBac…Eve…			0x5555556161bc
3	GThreadStd::run()			0x5555555f9476
4	??			0x7ffff6143d84
5	start_thread	pthread_create.c	463	0x7ffff6257590
6	clone	clone.S	95	0x7ffff5e30223

Number	Funct	File	Line	Address	Condition	Ignore	Threads

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

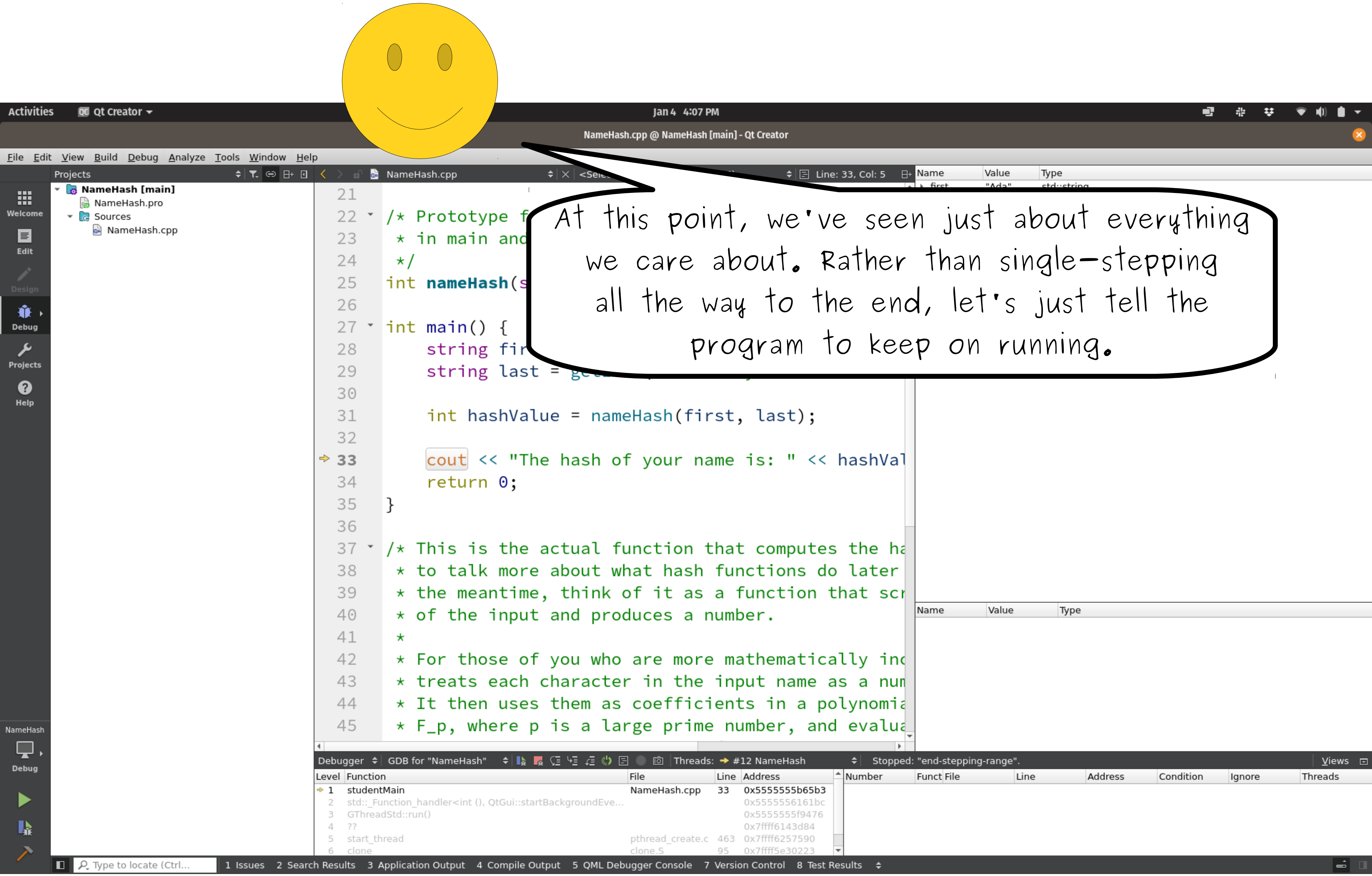
```
/* ...otype fo          nameHash function. This lets u
 * in main and then de              in the program.
 */
int nameHash(s

int main() {
    string fir
    string las

    int hashValue = nameHash(first, last);

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

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a num
 * It then uses them as coefficients in a polynomia
 * F_p, where p is a large prime number, and evalua
```

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

Name	Value	Type

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

Number	Funct	File	Line	Address	Condition	Ignore	Threads



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]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
Line: 33, Col: 5
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

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.

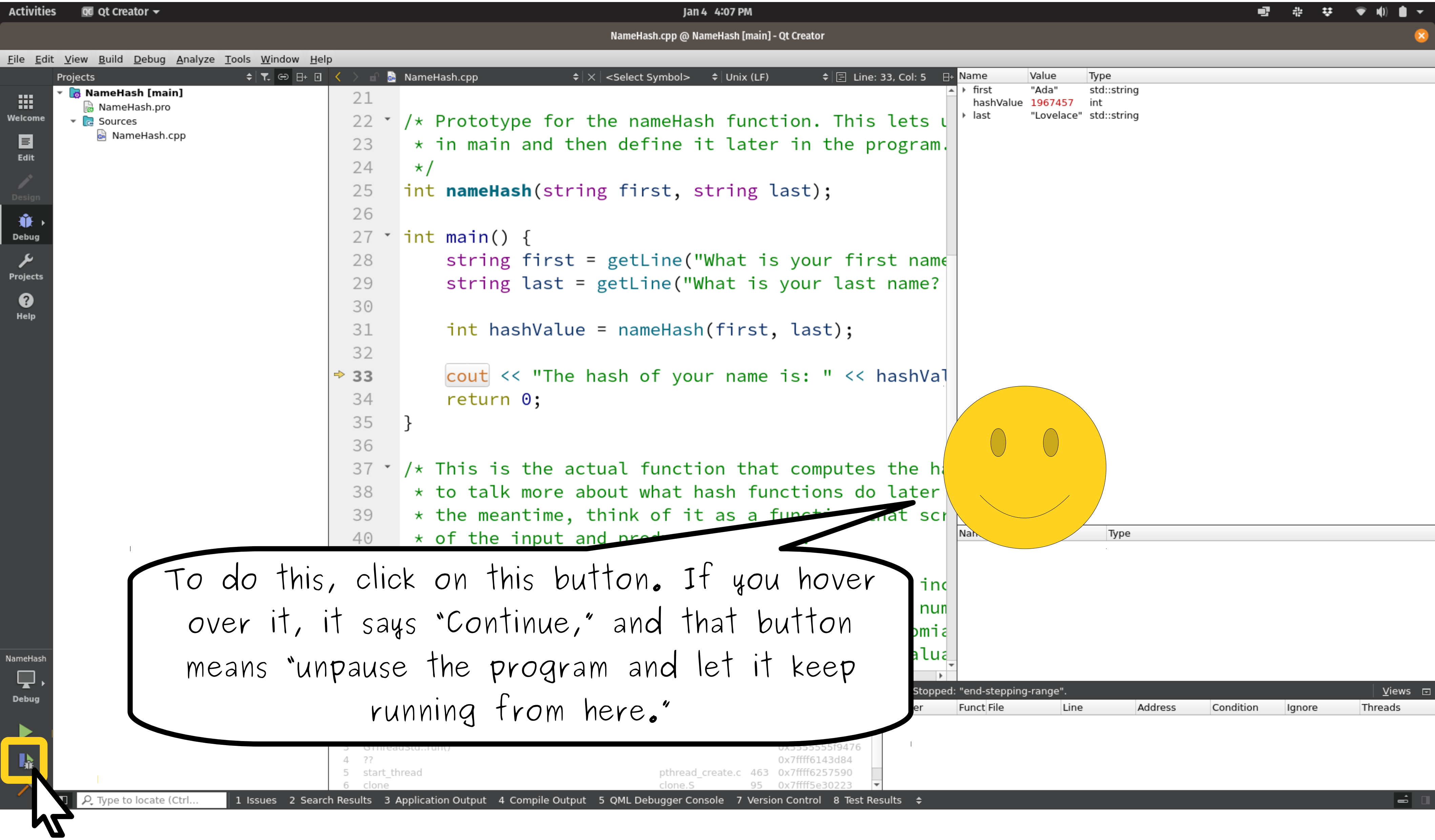
```
/* Prototype f
 * in main and
 */
int nameHash(s

int main() {
    string fir
    string last = ge

    int hashValue = nameHash(first, last);

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

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a function that scr
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inc
 * treats each character in the input name as a num
 * It then uses them as coefficients in a polynomia
 * F_p, where p is a large prime number, and evalua
```

Name	Value	Type
first	"Ada"	std::string

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

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

Number	Funct	File	Line	Address	Condition	Ignore	Threads


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]
NameHash.pro
Sources
NameHash.cpp
Welcome
Edit
Design
Debug
Projects
Help
NameHash
Debug
<Select Symbol>
Unix (LF)
Line: 33, Col: 5
Views
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

```
/* Prototype for the nameHash function. This lets u
 * 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: " << hashVal
    return 0;
}

/* This is the actual function that computes the ha
 * to talk more about what hash functions do later
 * the meantime, think of it as a functi      hat scr
 * of the input and pro
```

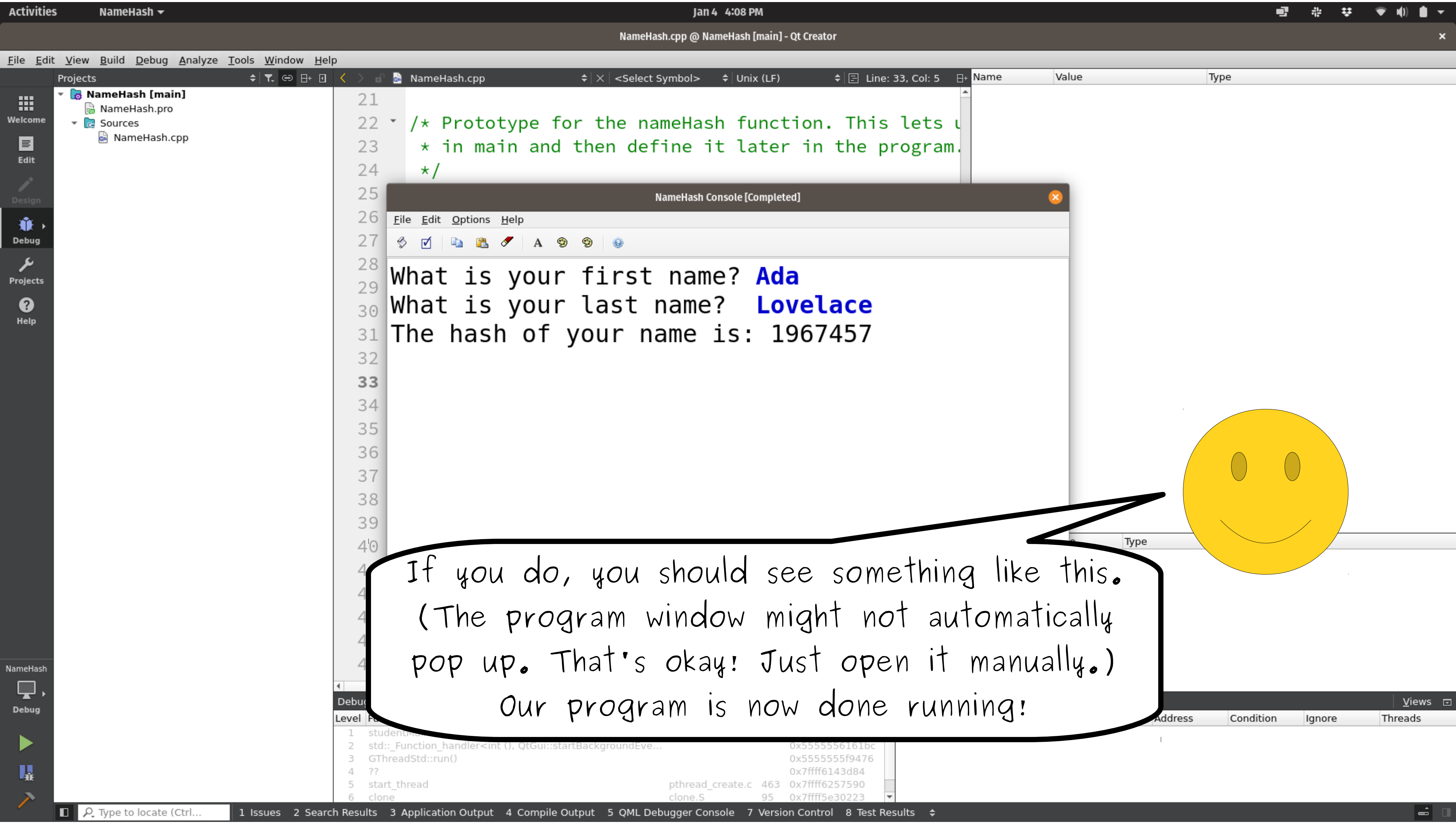
Name	Value	Type
first	"Ada"	std::string
hashValue	1967457	int
last	"Lovelace"	std::string

Stopped: "end-stepping-range".

er	Funct	File	Line	Address	Condition	Ignore	Threads

```
4  ??                                          0x7ffff6143d84
5  start_thread          pthread_create.c 463  0x7ffff6257590
6  clone                 clone.S          95   0x7ffff5e30223
```

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."

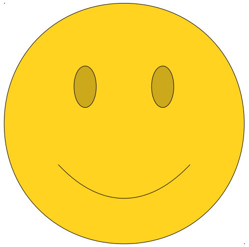

Activities
NameHash
Jan 4 4:08 PM
NameHash.cpp @ NameHash [main] - Qt Creator
File Edit View Build Debug Analyze Tools Window Help
Projects
NameHash [main]
NameHash.pro
Sources
NameHash.cpp
Line: 33, Col: 5
Name Value Type
NameHash Console [Completed]
File Edit Options Help
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 7 Version Control 8 Test Results

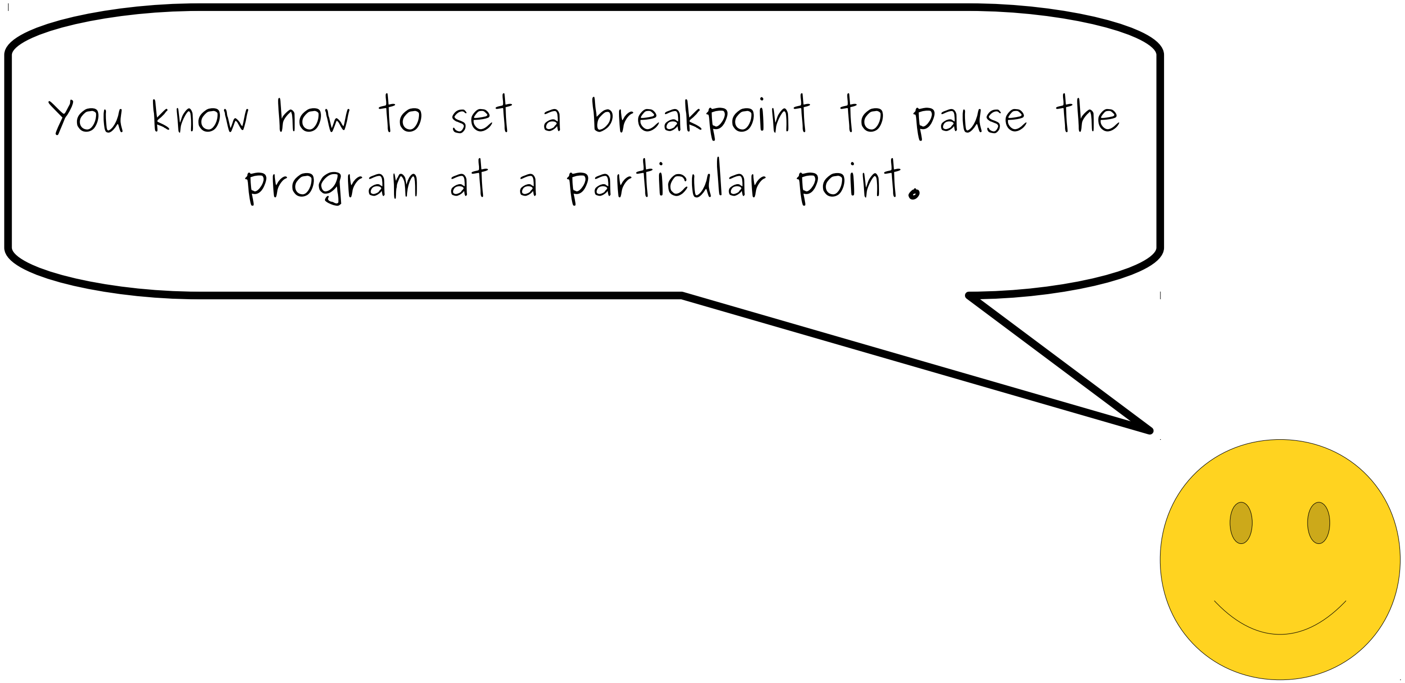
```
/* Prototype for the nameHash function. This lets u
 * in main and then define it later in the program.
 */
```

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

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!

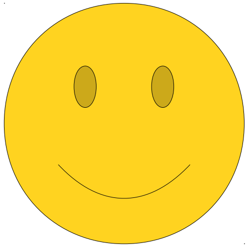
So there you have it! You've now gotten more familiar with the debugger!





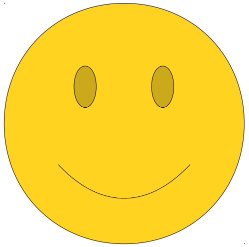
You know how to set a breakpoint to pause the program at a particular point.

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

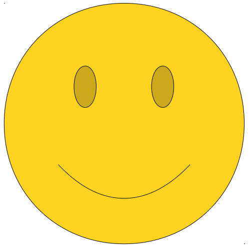


You know how to single-step the program and see what values change.

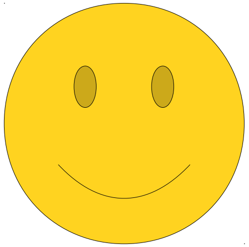
You know how to run a function to completion, and how to let the program keep on running.



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!