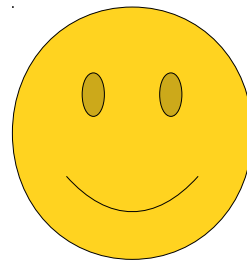
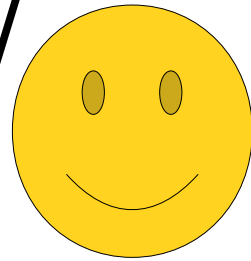


CS106B Debugger Tutorial

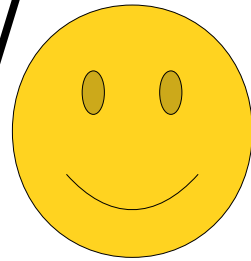
Hi everybody!



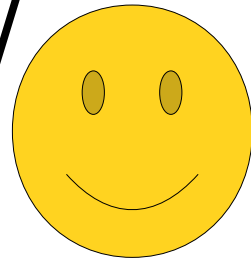
As part of Assignment 0, we'd like you to get a little bit of practice using the debugger in Qt Creator.



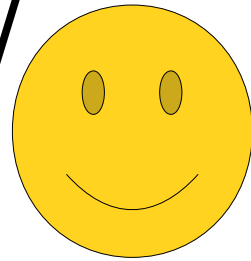
The debugger is a tool you can use to help see what your program is doing as you run it.



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.



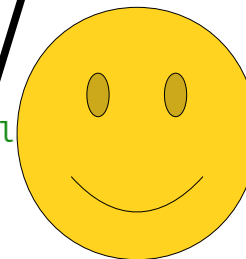
Think of this guide as a little tutorial walkthrough to help give you a sense of how to use the debugger and how to make sense of what you're seeing.





```
42 * For those of you who
43 * treats each character as a
44 * It then takes the first
45 * F_p, where p is a prime
46 * some small prime, and
47 * but we treat the first
48 */
49 int nameHash(const string& first, const string& last) {
50     /* This function computes the hash of a name.
51     * prime is a small prime number.
52     * 2^31 - 1 is a large prime number.
53     */
54     static const int kLargePrime = 15485863;
55     static const int kSmallPrime = 137;
56
57     int hashVal = 0;
58
59     /* Iterate across all the characters in the first name, then the last
60     * name, updating the hash at each step.
61     */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The numeric values of
64         * lower-case letters are always less than 127.
65         */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

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.



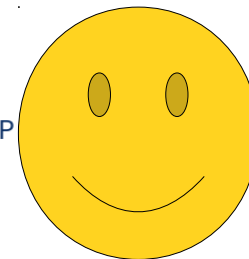
```
42 * For those of you who are more mathematically inclined, this function
43 * treats each character in the input name as a number between 0 and 128.
44 * It then uses them as coefficients in a polynomial over the finite field
45 *  $F_p$ , where  $p$  is a large prime number, and evaluates that polynomial at
46 * some smaller prime number  $q$ . (You aren't expected to know this for CS106B,
47 * but we thought it might be fun!)
48 */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers, a large prime and a small
51     * prime. These numbers were chosen because their product is less than
```

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

Now, click the mouse!

the last

```
62     for (char ch: first + last) {
63         /* Convert the input character to
64         * lower-case letters are always less than 128.
65         */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargeP
68     }
69
70     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
46 *  $x$ . See this for CS106B,

```

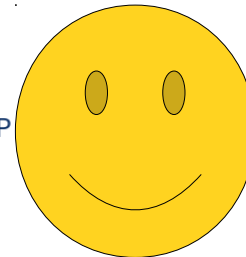
When you do, you should see a red circle with a little hourglass pop up.

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.

```

61
62 for (char ch: first + last) {
63     /* Convert the input character to a numeric value. The numeric values of
64     * lower-case letters are always less than 26.
65     */
66     ch = tolower(ch);
67     hashVal = (kSmallPrime * hashVal + ch) % kLargeP;
68 }
69
70 return hashVal;

```



Projects

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

Welcome

Edit

Design

Debug

Projects

Help

NameHash

Debug

```

42  * For those of you who are more mathematically inclined, this function
43  * treats each character in the input name as a number between 0 and 128.
44  * It then uses them as coefficients in a polynomial over the finite field
45  * F_p, where p is a large prime number, and evaluates that polynomial at
46  * some smaller prime number q. (You aren't expected to know this for CS106B,
47  * but we thought it might be fun!)
48  */
49  int nameHash(string first, string last){
50  /* This hashing scheme needs two prime numbers, a large prime and a small
51  * prime. These numbers were chosen because their product is less than

```

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

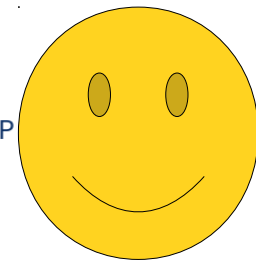
the last

```

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

```

The numeric values of

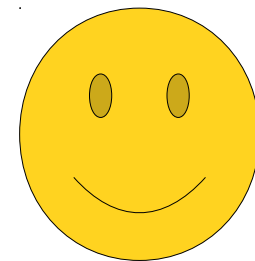


Projects

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

```
42 * For th
43 * treats
44 * It the
45 * F_p, v
46 * some s
47 * but we
48
49
50
51 What is your first name? |
52
53
54
55
56
57
58
59
60
61
62
63
64 * lower-case letters are always less than 127.
65 */
```

... 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.



Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
					1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3			(all)

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

NameHash

Debug

```

42 * For th
43 * treats
44 * It the
45 * F_p, v
46 * some s
47 * but we
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64 * lower-case letters are always less than 127.
65 */

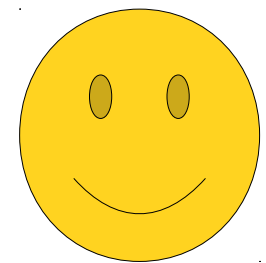
```

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.)

```

File Edit Options Help
What is your first name? Ada
What is your last name? |

```



Debugger GDB for "NameHash" Application started.

Level	Function	File	Line	Address	Con	Ignore	Threads
• 1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3			(all)

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

```

42 * For th
43 * treats
44 * It the
45 * F_p, v
46 * some s
47 * but we
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64 * lower-case letters are always less than 127.
65 */

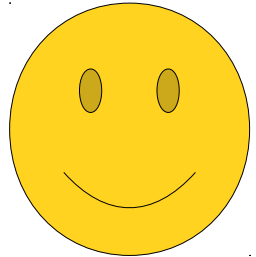
```

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

```

File Edit Options Help
What is your first name? Ada
What is your last name? Lovelace

```



roduc

ame, then

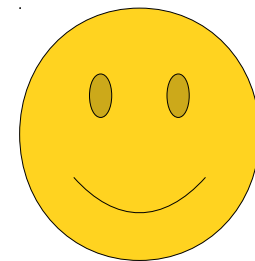
he numeric

Debugger GDB for "NameHash" Application started.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
					1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3			(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.

```
42 * For th
43 * treats
44 * It the
45 * F_p, v
46 * some s
47 * but we
48
49 File Edit Options Help
50
51 What is your first name? Ada
52 What is your last name? Lovelace
53
54
55
56
57
58
59
60
61
62
63
64 * lower-case letters are always less than 127.
65 */
```



Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
					1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3			(all)

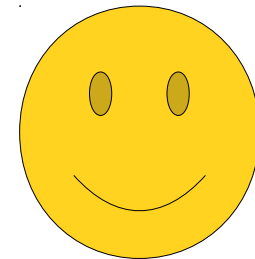
NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```
42 * For th  
43 * treats  
44 * It the  
45 * F_p, v  
46 * some s  
47 * but we
```

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

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

```
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64 * lower-case letters are always less than 127.  
65 */
```



```
ame, then
```

```
he numeric
```

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

Level	Function	File	Line	Address	Con/Ignore	Threads
• 1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)

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

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashV
58
59  /* Itera
60  * name
61  */
62  for (cha
63  /* C
64  * T
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
for_begin	@0x7ffffb2ffc78	std::string::iter...
for_end	@0x7ffffb2ffc80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3	(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb						
3	std::_Function_ha...			0x55555556037fc						
4	GThreadStd::run()			0x5555555e6616						
5	??			0x7ffff64dc2b3						
6	start_thread	pthread_create.c	442	0x7ffff6094b43						
7	clone3	clone3.S	81	0x7ffff6126a00						

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

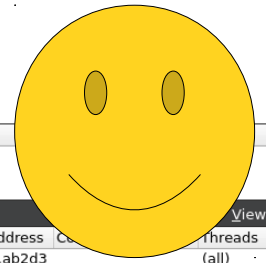
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashV
58
59  /* Itera
60  * name
61  */
62  for (ch
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70
71  return hashVal;
72

```

Name	Value	Type
__for_begin	@0x7fffb2ffc7b8	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



NameHash

Debug

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3	(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb						
3	std::_Function_ha...			0x55555556037fc						
4	GThreadStd::run()			0x5555555e6616						
5	??			0x7ffff64dc2b3						
6	start_thread	pthread_create.c	442	0x7ffff6094b43						
7	clone3	clone3.S	81	0x7ffff6126a00						

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 7 Version Control 8 Test Results

Projects NameHash.cpp Unix (LF) Line: 66, Col: 9

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashV
58
59  /* Itera
60  * name
61  */
62  for (ch
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70
71  return hashVal;
72

```

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb					
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

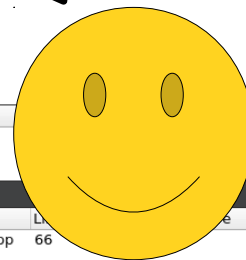
Debugger Variables:

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Debugger Console:

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 7 Version Control 8 Test Results

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



Projects NameHash.cpp Unix (LF) Line: 66, Col: 9

```


50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashV
58
59  /* Itera
60  * name
61  */
62  for (ch
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70
71  return hashVal;
72

```

Name Value Type

__for_begin	@0x7fff2fcb78	std::string::iter...
__for_end	@0x7fff2fcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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.



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...	66	(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...		
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

Projects | NameHash.cpp | nameHash(string, string) -> int | Unix (LF) | Line: 66, Col: 9

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashV
58
59  /* Itera
60  * name
61  */
62  for (ch
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70
71  return hashVal;
72

```


Debugger | GDB for "NameHash" | Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Views
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...	66	Threads (all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...		
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

Debugger Variables:

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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.



Projects

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

Welcome

Edit

Design

Debug

Projects

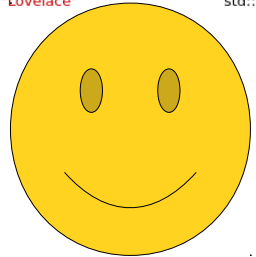
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71
72  }

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

```

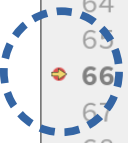
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = t
67      hashVa
68  }
69
70  return has
71
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLoveIace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"LoveIace"	std::string



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



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

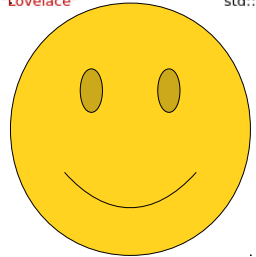
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



The call stack can tell us exactly that!

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

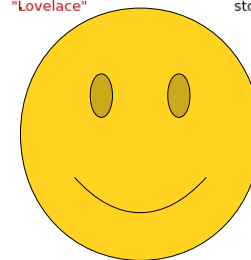
NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = t
67      hashVal
68  }
69
70  return has
71  }
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

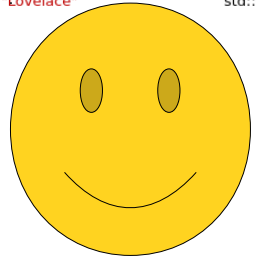
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71 }
72

```

Name	Value	Type
__for_begin	@0x7fff2fcb78	std::string::iter...
__for_end	@0x7fff2fcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

NameHash

Debug

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

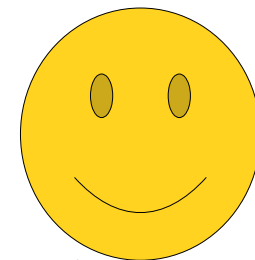
Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x55555550203	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	clone3	clone3.S	81	0x7ffff6126a00							
4	GThreadStd::ru			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							

```

Projects
  NameHash.cpp
  NameHash [main]
    NameHash.pro
    Sources
      NameHash.cpp
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
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: ";
34     return 0;
35 }
36
37 /* This is the actual function that
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 mathematically inclined, this f
43 * treats each character in the input name as a number between 0

```

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



You'll end up over here!

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64d2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects NameHash.cpp Unix (LF) Line: 31, Col: 5

```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
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: ";
34     return 0;
35 }
36
37 /* This is the actual function that
38 * to talk more about what hash func
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 math
43 * treats each character in the input

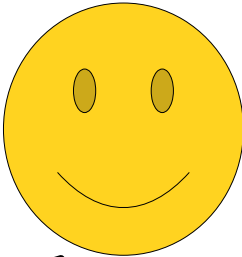
```

Debugger GDB for "NameHash" Threads: #15 NameHash

Level	Function	File	Line	Address	Number	Function
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::s...
2	studentMain	NameHash.cpp	31	0x5555555ab0fb		
3	std::_Function_ha...			0x5555556037fc		
4	GThreadStd::run()			0x5555555e6616		
5	??			0x7ffff64dc2b3		
6	start_thread	pthread_create.c	442	0x7ffff6094b43		
7	clone3	clone3.S	81	0x7ffff6126a00		

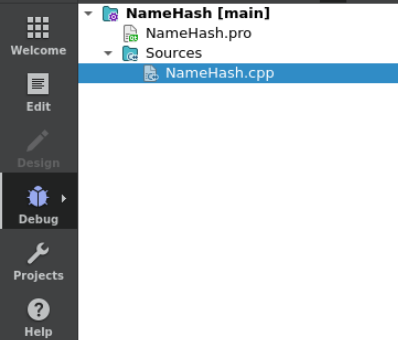
Debugger Variables:

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



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!

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 7 Version Control 8 Test Results

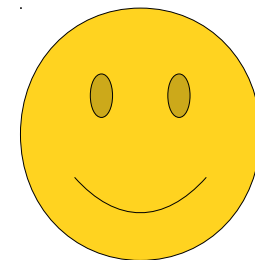


```

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

```

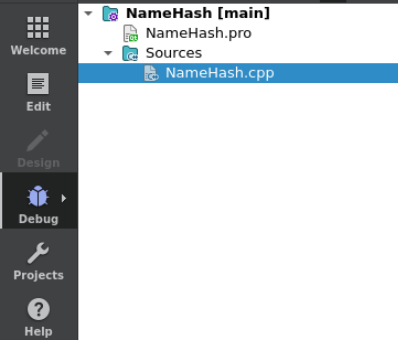
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string



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

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

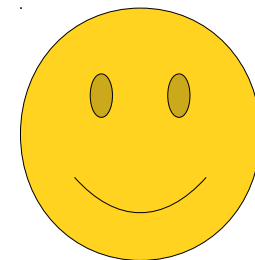


```

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

```

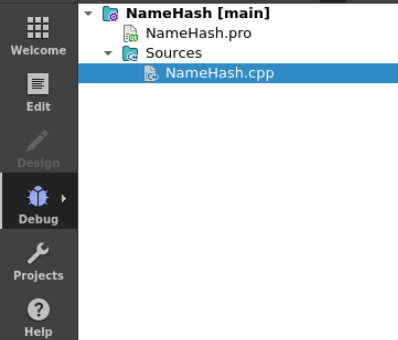
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string



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

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

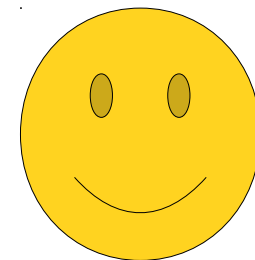


```

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

```

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



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

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

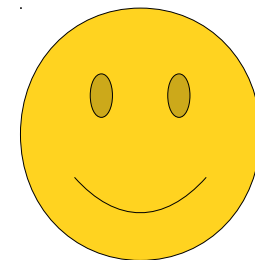
- NameHash [main]
 - NameHash.pro
 - Sources
 - NameHash.cpp

```

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

```

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

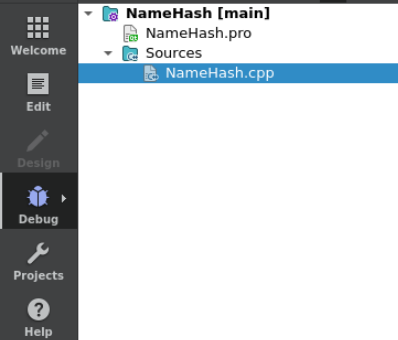


You don't need to worry about these. They'll show up in all the programs you run and you can safely ignore them.

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

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...	66	...		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...	66	...		(all)
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

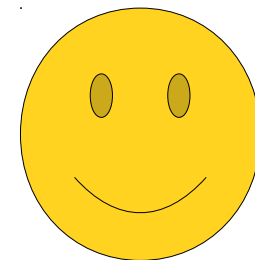


```

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

```

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



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

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	main	NameHash.cpp	31	0x5555555ab0fb							
3	std::Function			0x55555556037fc							
4	GThreadStd::r			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	thread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							



Projects

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

Welcome

Edit

Design

Debug

Projects

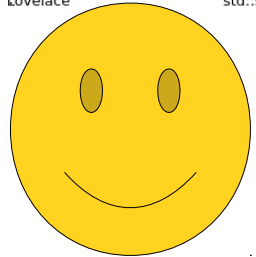
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The
64      * lower
65      */
66      ch = toLower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71 }
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



You'll be teleported back here!

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects: NameHash.cpp | nameHash(string, string) -> int | Unix (LF) | Line: 66, Col: 9

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate
60  * name, u
61  */
62  for (char
63  /* Con
64  * lowe
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71
72

```

Debugger: GDB for "NameHash" | Stopped at breakpoint 1 in thread 15.

Name	Value	Type
for_begin	@0x7fff2ffcb78	std::string::iter...
for_end	@0x7fff2ffcb80	std::string::iter...
for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Debugger Stack:

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x555555ab0fb							
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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



Projects NameHash.cpp Unix (LF) Line: 66, Col: 9

```


50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate
60  * name, u
61  */
62  for (char
63  /* Con
64  * lowe
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Debugger Variables:

Name	Value	Type
__for_begin	@0x7fff2ffc78	std::string::iter...
__for_end	@0x7fff2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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.



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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

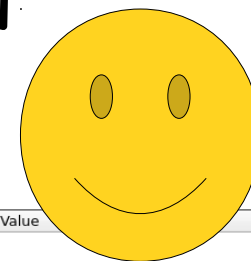
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal
58
59  /* Itera
60  * name
61  */
62  for (cha
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70
71  return hashVal;
72

```

Name	Value	Type
__for_begin	@0x7fff62ffc878	std::string::iter...
__for_end	@0x7fff62ffc880	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects NameHash.cpp Unix (LF) Line: 66, Col: 9

```

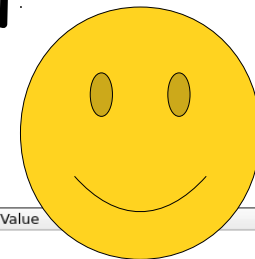
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal
58
59  /* Itera
60  * name
61  */
62  for (cha
63
64  /* C
65  * T
66  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Debugger: GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

The yellow arrow points out where we are right now.



NameHash [main]
NameHash.pro
Sources
NameHash.cpp

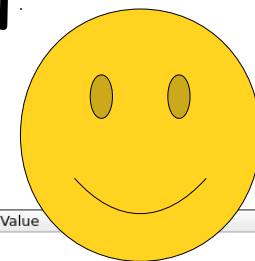
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal
58
59  /* Itera
60  * name
61  */
62  for (cha
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70  return hashVal;
71
72

```

Name	Value	Type
for_begin	@0x7fff2ffcb78	std::string::iter...
for_end	@0x7fff2ffcb80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555ab0fb							
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

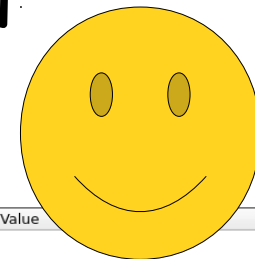
NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal
58
59  /* Itera
60  * name
61  */
62  for (cha
63
64  /* C
65  * T
66  */
67  ch = tolower(ch);
68  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70  return hashVal;
71
72

```

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



Name	Value	Type
for_begin	@0x7fff62ffc878	std::string::iter...
for_end	@0x7fff62ffc880	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							



Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

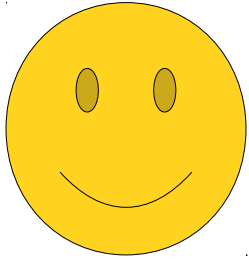
```

58     int hashVal = 0;
59
60     /* Iterate across all the characters in the first name, then
61      * name, updating the hash at each step.
62      */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numeric
65          * lower-case letters are always less than 127.
66          */
67         ch = tolower(ch);
68         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69     }
70
71     return hashVal;
72 }

```

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

This window lets you take a look at all the values of the local variables that are in scope now. (Don't worry if you see different values or "not accessible" on your system - that's okay!)



```
58     int hashVal = 0;
59
60     /* Iterate across all the characters in the first name, then
61      * name, updating the hash at each step.
62      */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numeric
65          * lower-case letters are always less than 127.
66          */
67         ch = tolower(ch);
68         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69     }
70
71     return hashVal;
72 }
```

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...				
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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`.



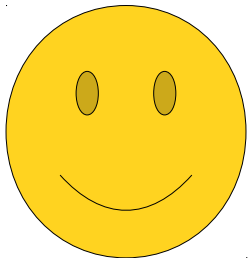
```
int hashVal = 0;
hashVal: 0
58
59 /* Iterate across all the characters in the first name, then
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) { first: "Ada" last: "Love...
63 /* Convert the input character to lower case. The numeric
64 * lower-case letters are always less than 127.
65 */
66 ch = tolower(ch); ch: 65
67 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68 }
69
70 return hashVal;
71 }
72
```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLoveIace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"LoveIace"	std::string

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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



```
int hashVal = 0;                                hashVal: 0
58
59 /* Iterate across all the characters in the first name, then
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {                  first: "Ada" last: "Love...
63     /* Convert the input character to lower case. The numeric
64     * lower-case letters are always less than 127.
65     */
66     ch = tolower(ch);                          ch: 65
67     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68 }
69
70 return hashVal;
71 }
72
```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...				
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

```

Projects
  NameHash [main]
    NameHash.pro
    Sources
      NameHash.cpp
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61
62  "Love...
63  numeric
64
65  ch: 65
66  time;
67
68  return hashVal;
69
70
71
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

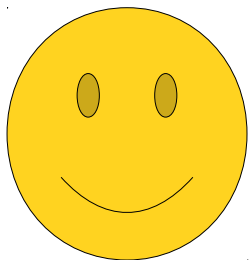
```

50
51 /* This hashing scheme needs two prime numbers, a large prime
52 * prime. These numbers were chosen because their product is
53 *  $2^{31} - kLargePrime - 1$ .
54 */
55 static const int kLargePrime = 15485863;
56 static const int kSmallPrime = 137;
57
58 int hashVal = 0;
59
60 /* Iterate across all the characters in the first name, then
61 * name, updating the hash at each step

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



```

71 return hashVal;
72

```

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

NameHash [main]

NameHash.pro

Sources

NameHash.cpp

50

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

51

* prime. These numbers were chosen because their product is

52

* $2^{31} - kLargePrime - 1$.

53

*/

54

static const int kLargePrime = 15485863; kLargePrime: 1...

55

static const int kSmallPrime = 137; kSmallPrime: 137

56

57

int hashVal = 0; hashVal: 0

58

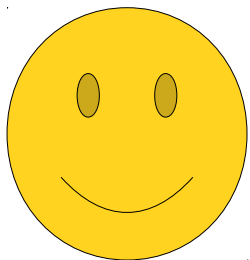
59

/* Iterate across all the characters in the first name, then

60

* name, updating the hash at each step

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



61

return hashVal;

62

63

64

65

66

67

68

69

70

71

72

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

"Love...

numeric

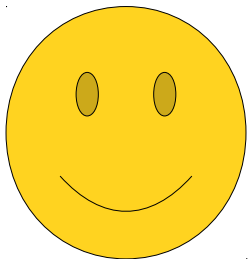
ch: 65

time;

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x555555ab0fb							
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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



```

hashVal = 0;

/* Iterate across all the characters in the first name, then
 * name, updating the hash at each step.
 */
for (char ch: first + last) { first: "Ada" last: "Love...
/* Convert the input character to lower case. The numeric
 * lower-case letters are always less than 127.
 */
ch = tolower(ch); ch: 65
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}

return hashVal;
}

```

Name	Value	Type
__for_begin	@0x7ffb2ffc78	std::string::iter...
__for_end	@0x7ffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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



```

65 hashVal = 0;
66
67 /* Iterate across all the characters in the first name, then
68 * name, updating the hash at each step.
69 */
70 for (char ch: first + last) { first: "Ada" last: "Love...
71     /* Convert the input character to lower case. The numeric
72     * lower-case letters are always less than 127.
73     */
74     ch = tolower(ch); ch: 65
75     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
76 }
77
78 return hashVal;
79
80 }
81
82 }

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Prime: 1...

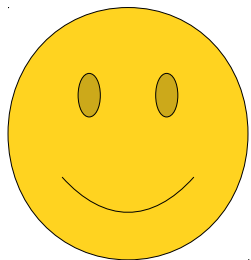
Prime: 137

hashVal: 0

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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



```

59
60
61
62 for (char ch: first + last) { first: "Ada" last: "Love...
63
64     /* Convert the input character to lower case. The numeric
65     * lower-case letters are always less than 127.
66     */
67     ch = tolower(ch); ch: 65
68     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69
70     return hashVal;
71 }
72

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Prime: 1...

Prime: 137

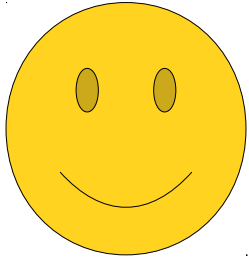
hashVal: 0

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

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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.



```
hashVal = 0;
```

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

```
for (char ch: first + last) { first: "Ada" last: "Love...
/* Convert the input character to lower case. The numeric
 * lower-case letters are always less than 127.
 */
ch = tolower(ch); ch: 65
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
```

```
return hashVal;
```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
for_range	"AdaLovelace"	std::string &&
ch	'A' 65	0x41 char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

```
Prime: 1...
```

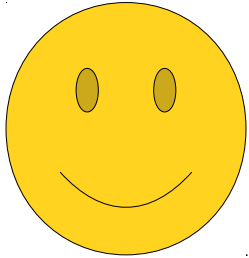
```
Prime: 137
```

```
hashVal: 0
```

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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.



```

59     hashVal = 0;
60
61     /* Iterate across all the characters in the first name, then
62      * name, updating the hash at each step.
63      */
64     for (char ch: first + last) { first: "Ada" last: "Love...
65         /* Convert the input character to lower case. The numeric
66         * lower-case letters are always less than 127.
67         */
68         ch = tolower(ch); ch: 65
69         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
70     }
71     return hashVal;
72 }

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	char 0x41
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Prime: 1...

Prime: 137

hashVal: 0

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

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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!



```

59     hashVal = 0;
60
61     /* Iterate across all the characters in the first name, then
62     * name, updating the hash at each step.
63     */
64     for (char ch: first + last) { first: "Ada" last: "Love...
65         /* Convert the input character to lower case. The numeric
66         * lower-case letters are always less than 127.
67         */
68         ch = tolower(ch); ch: 65
69         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
70     }
71     return hashVal;
72 }

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'A' 65	0x41 char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

kLargePrime: 15485863

kSmallPrime: 137

hashVal: 0

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects NameHash.cpp Unix (LF) Line: 66, Col: 9

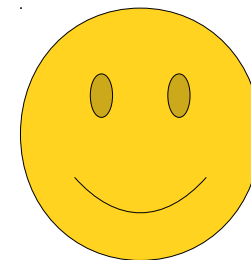
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71 }
72

```

Debugger: GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3			(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb								
3	std::Function_ha...			0x55555556037fc								
4	GThreadStd::run()			0x5555555e6616								
5	??			0x7ffff64dc2b3								
6	start_thread	pthread_create.c	442	0x7ffff6094b43								
7	clone3	clone3.S	81	0x7ffff6126a00								



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

Projects

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

```

50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72

```

```

/* This hashing scheme needs two prime numbers, a large prime
 * prime. These numbers were chosen because their product is
 *  $2^{31} - kLargePrime - 1$ .
 */
static const int kLargePrime = 15485863;
static const int kSmallPrime = 137;

int hashVal = 0;

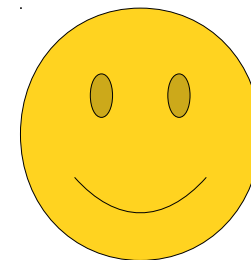
/* Iterate across all the characters in the first name, then
 * name, updating the hash at each step.
 */
for (char ch: first + last) {
    /* Convert the input character to lower case. The numeric
     * lower-case letters are always less than 127.
     */
    ch
    has

}

return
}

```

Name	Value	Type
__for_begin	@0x7fffb2ffcb78	std::string::iter...
__for_end	@0x7fffb2ffcb80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped at breakpoint 1 in thread 15. Views

Level	Function	File	Number	Function	File	Line	Address	Con. Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	nameHash(std::string, std::string)				(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb					
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

Projects

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

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

```

/* This hashing scheme needs two prime numbers, a large prime
 * prime. These numbers were chosen because their product is
 *  $2^{31} - kLargePrime - 1$ .
 */
static const int kLargePrime = 15485863;
static const int kSmallPrime = 137;

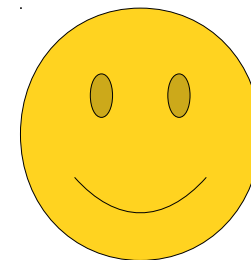
int hashVal = 0;

/* Iterate across all the characters in the first name, then
 * name, updating the hash at each step.
 */
for (char ch: first + last) {
    /* Convert the input character to lower case. The numeric
     * lower-case letters are always less than 127.
     */
    ch
    has
}

return
}

```

Name	Value	Type
__for_begin	@0x7fffb2ffcb78	std::string::iter...
__for_end	@0x7fffb2ffcb80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con. Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...				
3	std::Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

```

/* This hashing scheme needs two prime numbers, a large prime
 * prime. These numbers were chosen because their product is
 *  $2^{31} - kLargePrime - 1$ .
 */
static const int kLargePrime = 15485863;
static const int kSmallPrime = 137;

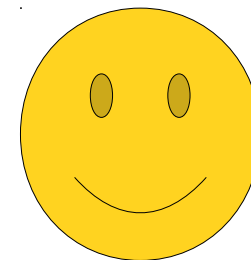
int hashVal = 0;

/* Iterate across all the characters in the first name, then
 * name, updating the hash at each step.
 */
for (char ch: first + last) {
    /* Convert the input character to lower case. The numeric
     * lower-case letters are always less than 127.
     */
    ch
    has
}

return
}

```

Name	Value	Type
__for_begin	@0x7fffb2ffcb78	std::string::iter...
__for_end	@0x7fffb2ffcb80	std::string::iter...
__for_range	"AdaLoveIace"	std::string &&
ch	'A' 65	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"LoveIace"	std::string



Once you're confident that you're on the "Step Over" button - and not the "Step Into" or "Step Out" buttons - go and click it! When you do...

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con. Ignore	Threads
1	nameHash	NameHash.cpp	66	...5555ab2d3	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	...5555ab0fb							
3	std::Function_ha...			...56037fc							
4	GThreadStd::run()			...5555e6616							
5	??			0x711m64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

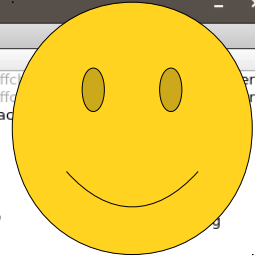
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate a
60  * name, up
61  */
62  for (char ch
63  /* Conve
64  * lower
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value
for_begin	@0x7fff2ffc
for_end	@0x7fff2ffc
for_range	"AdaLovelad
ch	'a'
first	"Ada"
hashVal	0
kLargePrime	15485863
kSmallPrime	137
last	"Lovelace"



...your window should look something like this.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

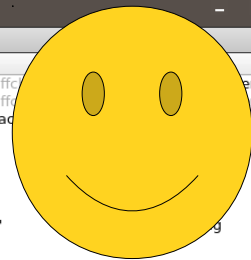
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate a
60  * name, up
61  */
62  for (char ch
63  /* Conve
64  * lower
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value
__for_begin	@0x7fff2ffc
__for_end	@0x7ffb2ff
__for_range	"AdaLovelac
ch	'a'
first	"Ada"
hashVal	0
kLargePrime	15485863
kSmallPrime	137
last	"Lovelace"



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

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects NameHash.cpp nameHash(string, string) -> int Unix (LF) Line: 67, Col: 9

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

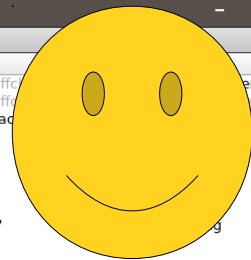
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate a
60  * name, up
61  */
62  for (char ch
63  /* Conve
64  * lower
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Debugger GDB for "NameHash" #15 NameHash Stopped: "end-stepping-range".

Name	Value
__for_begin	@0x7fff2ffc
__for_end	@0x7fff2ffc
__for_range	"AdaLovela
ch	'a'
first	"Ada"
hashVal	0
kLargePrime	15485863
kSmallPrime	137
last	"Lovela"

Name Value Type



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

Projects | NameHash.cpp | Line: 67, Col: 9

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate a
60  * name, up
61  */
62  for (char ch
63  /* Conve
64  * lower
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

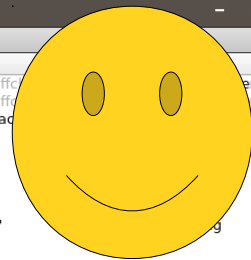
```

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

Name	Value	Type
__for_begin	@0x7ffffb2ffc	
__for_end	@0x7ffffb2ffc	
__for_range	"AdaLovela	
ch	'a'	
first	"Ada"	
hashVal	0	
kLargePrime	15485863	
kSmallPrime	137	
last	"Lovela"	

Debugger | Level | Function | File | Line | Address | Number | Function | File | Line | Address | Con | Ignore | Threads

1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	eHash.cpp	66	...ab2d3			(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb									
3	std::_Function_ha...			0x55555556037fc									
4	GThreadStd::run()			0x5555555e6616									
5	??			0x7ffff64dc2b3									
6	start_thread	pthread_create.c	442	0x7ffff6094b43									
7	clone3	clone3.S	81	0x7ffff6126a00									



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

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

NameHash

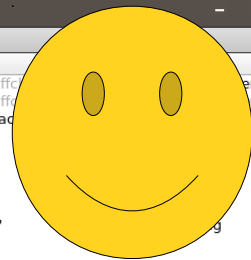
Debug

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate a
60  * name, up
61  */
62  for (char ch
63  /* Conve
64  * lower
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

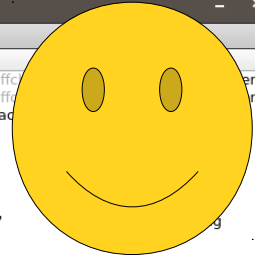
Name	Value
__for_begin	@0x7fff2ffc
__for_end	@0x7fff2ffc
__for_range	"AdaLovela
ch	'a'
first	"Ada"
hashVal	0
kLargePrime	15485863
kSmallPrime	137
last	"Lovela"



so what did that line of code do?

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037c							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							



Projects

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

Debugger

Debug

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate a
60  * name, up
61  */
62  for (char ch
63  /* Conve
64  * lower
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value
__for_begin	@0x7fff2ffc
__for_end	@0x7fff2ffc
__for_range	"AdaLovela
ch	'a'
first	"Ada"
hashVal	0
kLargePrime	15485863
kSmallPrime	137
last	"Lovela"

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.

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037c							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

 NameHash [ma...
 NameHash.p...
 Sources
 NameHas...

Welcome

Edit

Design

Debug

Projects

Help

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



```

59  */ Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
▶ _for_begin	@0x7fffb2ffc78	std::string::iter...
▶ _for_end	@0x7fffb2ffc80	std::string::iter...
▶ _for_range	"AdaLovelace"	std::string &&
ch	'a' 97	0x61 char
▶ first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
▶ last	"Lovelace"	std::string

NameHash

Debug

Debugger ▾ GDB for "NameHash" ▾ #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
▶ 1	nameHash	NameHash.cpp	67	0x555555ab2e1	• 1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x555555ab0fb							
3	std::_Function_ha...			0x5555556037fc							
4	GThreadStd::run()			0x555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

NameHash [ma
NameHash.p
Sources
NameHas

Welcome

Edit

Design

Debug

Projects

Help

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



```

59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
for_begin	@0x7fffb2ffc78	std::string::iter...
for_end	@0x7fffb2ffc80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Debugger GDB for "NameHash" #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

NameHash [ma
NameHash.p
Sources
NameHas

Welcome

Edit

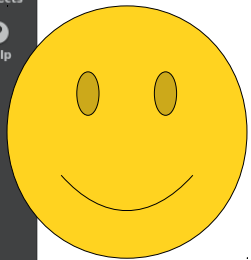
Design

Debug

Projects

Help

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



```

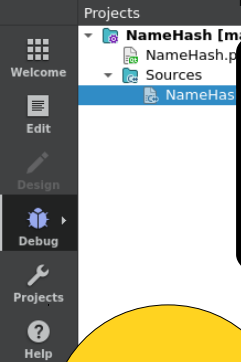
59  */ Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
for_begin	@0x7fffb2ffc78	std::string::iter...
for_end	@0x7fffb2ffc80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							



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.



```

59
60
61
62 for (char ch: first + last) { first: "Ada" last: "Love..."
63     /* Convert the input character to lower case. The numeric
64     * lower-case letters are always less than 127.
65     */
66     ch = tolower(ch);
67     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68 }
69
70 return hashVal;
71 }
72

```

Name	Value	Type
for_begin	@0x7fffb2ffc78	std::string::iter...
for_end	@0x7fffb2ffc80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'a'	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

NameHash [ma
NameHash.p
Sources
NameHas

Welcome

Edit

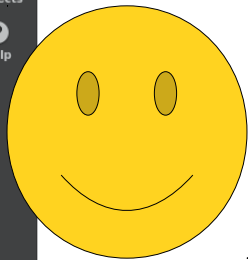
Design

Debug

Projects

Help

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



```

59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
▶ __for_begin	@0x7fffb2ffc78	std::string::iter...
▶ __for_end	@0x7fffb2ffc80	std::string::iter...
▶ __for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
▶ last	"Lovelace"	std::string

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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
▶ 1	nameHash	NameHash.cpp	67	0x5555555ab2e1	• 1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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.



```

59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char 0x61
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

NameHash [ma
NameHash.p
Sources
NameHas

Welcome

Edit

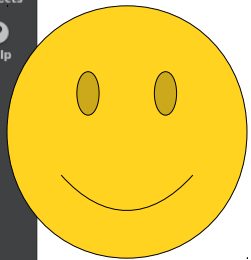
Design

Debug

Projects

Help

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



```

59  */ Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
__for_begin	@0x7ffb2ffc78	std::string::iter...
__for_end	@0x7ffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037c							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

NameHash [ma
NameHash.p
Sources
NameHas

Welcome

Edit

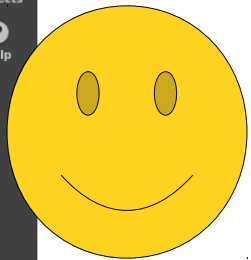
Design

Debug

Projects

Help

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



```

59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Love...
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Large prime
product is

Name	Value	Type
▶ __for_begin	@0x7fffb2ffc78	std::string::iter...
▶ __for_end	@0x7fffb2ffc80	std::string::iter...
▶ __for_range	"AdaLovelace"	std::string &&
ch	'a' 97	0x61 char
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
▶ last	"Lovelace"	std::string

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
▶ 1	nameHash	NameHash.cpp	67	0x5555555ab2e1	• 1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects NameHash.cpp Unix (LF) Line: 67, Col: 9

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (int i = 0; i < first_name.length(); ++i) {
63
64
65
66
67
68
69
70
71
72

```

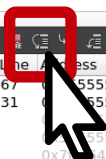
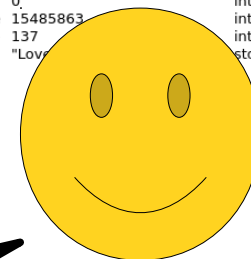
Debugger Variables:

Name	Value	Type
__for_begin	@0x7ffb2ffc78	std::string::iter...
__for_end	@0x7ffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char 0x61
first	"Ada"	std::string
hashVal	0	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Debugger:

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	...5555ab2e1	1	nameHash(std::string, std::string)	...				
2	studentMain	NameHash.cpp	31	...5555ab0fb			...eHash.cpp	66	...ab2d3		(all)
3	std::_Function_ha...			...56037fc							
4	GThreadStd::run()			...5555e6616							
5	??			...4dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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



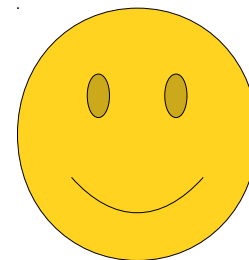
NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char 0x61
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

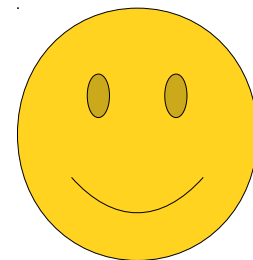
Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con. Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

```

Projects
  NameHash.cpp
  NameHash [main]
  NameHash.pro
  Sources
    NameHash.cpp
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

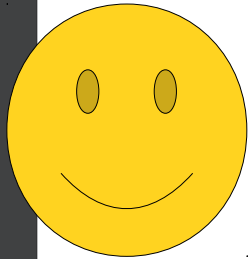
Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char 0x61
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



Let's see what's changed.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b	1	nameHash(std::string, std::string)	...eHash.cpp	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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!



```

61
62 → for (char ch: first + last) { ch: 97 first: "Ada" las...
63
64 /* Convert the input character to lower case. The numeric
65 * lower-case letters are always less than 127.
66 */
66 ch = tolower(ch);
67 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68 }
69
70 return hashVal;
71 }
72

```

Name	Value	Type
▶ _for_begin	@0x7fffb2ffc878	std::string::iter...
▶ _for_end	@0x7fffb2ffc880	std::string::iter...
▶ _for_range	"AdaLovelace"	std::string &&
ch	'a'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
▶ last	"Lovelace"	std::string

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
▶ 1	nameHash	NameHash.cpp	62	0x5555555ab31b	• 1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

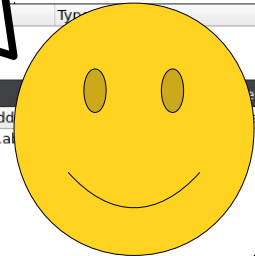
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch = *name; ch != '\0'; ++ch)
63  {
64      /* Con
65      * low
66      */
66      ch = t
67      hashVal
68  }
69
70  return hashVal;
71
72

```

Name	Value	Type
for_begin	@0x7fff2fcb78	std::string::iter...
for_end	@0x7fff2fcb80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'a'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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.



NameHash

Debug

Debugger GDB for "NameHash" #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Add
1	nameHash	NameHash.cpp	62	0x555555ab31b	1	nameHash(std::string, std::string)	...	66	...
2	studentMain	NameHash.cpp	31	0x555555ab0fb					
3	std::_Function_ha...			0x5555556037fc					
4	GThreadStd::run()			0x555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

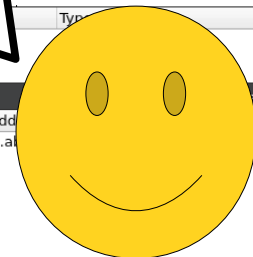
```

Projects
  NameHash.cpp
  NameHash [main]
  NameHash.pro
  Sources
    NameHash.cpp
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71 }
72

```

Name	Value	Type
▶ _for_begin	@0x7fff2fcb78	std::string::iter...
▶ _for_end	@0x7fff2fcb80	std::string::iter...
▶ _for_range	"AdaLovelace"	std::string &&
ch	'a' 97	char 0x61
▶ first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
▶ last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Add
▶ 1	nameHash	NameHash.cpp	62	0x5555555ab31b	• 1	nameHash(std::string, std::string)	...	66	...
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...		
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

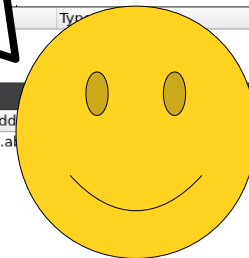
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'a' 97	char 0x61
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

Let's go do it again!



Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Add
1	nameHash	NameHash.cpp	62	0x5555555ab31b	1	nameHash(std::string, std::string)	...	66	...
2	studentMain	NameHash.cpp	31	0x5555555ab0fb		
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

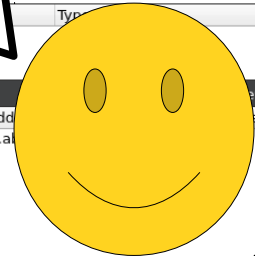
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = toLower(ch);
67      hashVal = (hashVal * kSmallPrime + ch) % kLargePrime;
68  }
69
70  return hashVal;
71 }
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'a'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



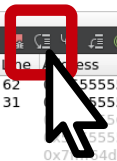
Debugger

GDB for "NameHash"

Threads: #15 NameHash

Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Add
1	nameHash	NameHash.cpp	62	0x55555ab31b	1	nameHash(std::string, std::string)	...eHash.cpp	66	...
2	studentMain	NameHash.cpp	31	0x5555ab0fb					
3	std::_Function_ha...			0x5556037fc					
4	GThreadStd::run()			0x5555e6616					
5	??			0x7ffffd4c2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					



Projects NameHash.cpp nameHash(string, string) -> int Unix (LF) Line: 66, Col: 9

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

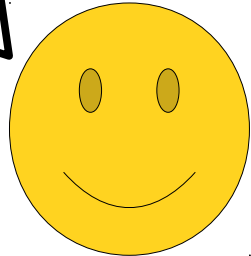
50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal
58
59  /* Iterate
60  * name, u
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Debugger Variables:

Name	Value	Type
for_begin	@0x7ffff2fcb78	std::string::iter...
for_end	@0x7ffff2fcb80	std::string::iter...
for_range	"Ada Lovelace"	std::string &&
ch	'd' 100	char 0x64
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	0x5555555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

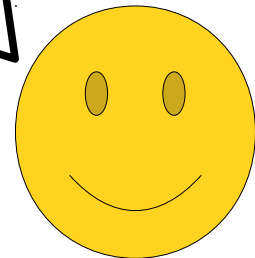
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal
58
59  /* Iterate
60  * name, u
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71
72

```

Name	Value	Type
__for_begin	@0x7fff2ffc78	std::string::iter...
__for_end	@0x7fff2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'd' 100	char 0x64
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



NameHash

Debug

Debugger GDB for "NameHash" Stopped at breakpoint 1 in thread 15.

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	66	...5555ab2d3	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	...5555ab0fb							
3	std::_Function_ha...			...56037fc							
4	GThreadStd::run()			...555e6616							
5	??			0x7ffff34dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							



Projects NameHash.cpp nameHash(string, string) -> int Unix (LF) Line: 67, Col: 9

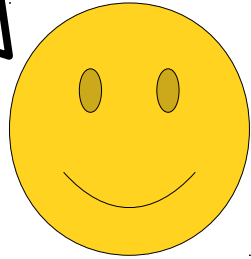
Name	Value	Type
for_begin	@0x7ffffb2ffc78	std::string::iter...
for_end	@0x7ffffb2ffc80	std::string::iter...
for_range	"AdaLovelace"	std::string &&
ch	'd' 100	0x64 char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate over the characters in the name.
60  * name, up to last.
61  */
62  for (char ch: first + last) { first: "Ada" last: "Lovelace"
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

You should be here now. Notice that none of the values changed. That makes sense, since all we did was convert a lower-case 'd' to a lower-case 'd'.



Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	67	0x5555555ab2e1	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

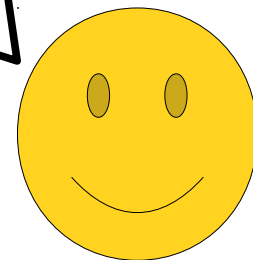
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal(const string& name) {
58
59  /* Iterate over the characters in the name.
60  * name, up to last.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case. The numeric
64      * lower-case letters are always less than 127.
65      */
66      ch = tolower(ch);
67      hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
▶ __for_begin	@0x7fff2ffcb78	std::string::iter...
▶ __for_end	@0x7fff2ffcb80	std::string::iter...
▶ __for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	97	int
kLargePrime	15485863	int
kSmallPrime	137	int
▶ last	"Lovelace"	std::string

Now, click "Step Over" one more time.



Debugger GDB for "NameHash" #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
▶ 1	nameHash	NameHash.cpp	67	...5555ab2e1	• 1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	...555ab0fb							
3	std::_Function_ha...			...6037fc							
4	GThreadStd::run()			...555e6616							
5	??			0x7fff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

```

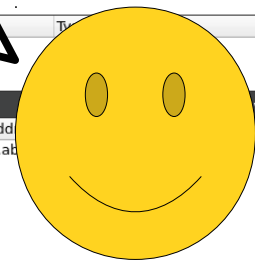
50  /* This hashing scheme needs two prime numbers, kLargePrime and kSmallPrime.
51  * prime. These numbers were chosen because:
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72

```

Look here!

Name	Value	Type
__for_begin	@0x7fffb2ffc878	std::string::iter...
__for_end	@0x7fffb2ffc880	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string

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



Debugger

GDB for "NameHash"

Threads: #15 NameHash

Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Add
1	nameHash	NameHash.cpp	62	0x5555555ab31b	1	nameHash(std::string, std::string)	...	66	...
2	studentMain	NameHash.cpp	31	0x5555555ab0fb					
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

Projects NameHash.cpp Unix (LF) Line: 62, Col: 5

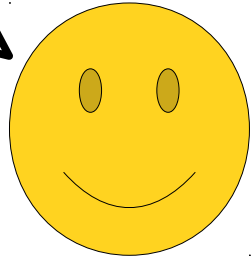
Name	Value	Type
__for_begin	@0x7fff2ffc78	std::string::iter...
__for_end	@0x7fff2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
		std::string

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54
55
56
57
58
59
60
61  */
62  for (char ch: first + last) {   ch: 100 first: "Ada
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71 }
72

```

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.



Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b	1	nameHash(std::string, std::string)	...	66	...ab2d3		(all)
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

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

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 7 Version Control 8 Test Results

Projects

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

Debugger

Level Function File Line Address

1	nameHash	NameHash.cpp	62	0x5555555ab31b
2	studentMain	NameHash.cpp	31	0x5555555ab0fb
3	std::_Function_ha...			0x55555556037fc
4	GThreadStd::run()			0x5555555e6616
5	??			0x7ffff64dc2b3
6	start_thread	pthread_create.c	442	0x7ffff6094b43
7	clone3	clone3.S	81	0x7ffff6126a00

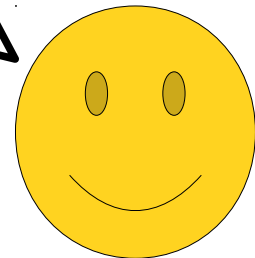
```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54
55
56
57
58
59
60
61  */
62  for (char ch: first + last) { ch: 100 first: "Ada
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68
69
70  return hashVal;
71
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffcb78	std::string::iter...
__for_end	@0x7ffffb2ffcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
		int
		std::string

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



Debugger GDB for "NameHash" #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b	1	nameHash(std::string, std::string)					
2	studentMain	NameHash.cpp	31	0x5555555ab0fb			...eHash.cpp	66	...ab2d3		(all)
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects NameHash.cpp Unix (LF) Line: 62, Col: 5

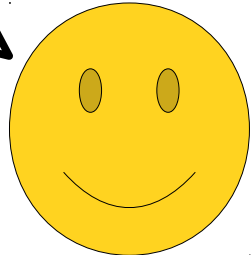
Name	Value	Type
__for_begin	@0x7fff2fcb78	std::string::iter...
__for_end	@0x7fff2fcb80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	???	int
kLargePrime	15485863	int
		std::string

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54
55
56
57
58
59
60
61  */
62  for (char ch: first + last) { ch: 100 first: "Ada
63  /* Convert the input character to lower case. The numeric
64  * lower-case letters are always less than 127.
65  */
66  ch = tolower(ch);
67  hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68  }
69
70  return hashVal;
71  }
72

```

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



Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b								
2	studentMain	NameHash.cpp	31	0x5555555ab0fb								
3	std::_Function_ha...			0x55555556037fc								
4	GThreadStd::run()			0x5555555e6616								
5	??			0x7ffff64dc2b3								
6	start_thread	pthread_create.c	442	0x7ffff6094b43								
7	clone3	clone3.S	81	0x7ffff6126a00								

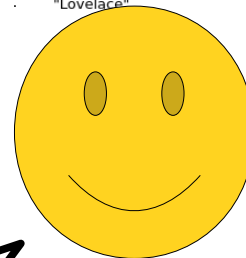
NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case
64       * lower
65       */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71  }
72

```

Name	Value	Type
for_begin	@0x7ffffb2ffc78	std::string::iter...
for_end	@0x7ffffb2ffc80	std::string::iter...
for_range	"AdaLovelace"	std::string &&
ch	'd' 100	char 0x64
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



Now, take a look back at these buttons.

Debugger GDB for "NameHash"



Threads: #15 NameHash

Stopped: "end-stepping-range".

Views

Level	Function	File	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b					
2	studentMain	NameHash.cpp	31	0x5555555ab0fb					
3	std::_Function_ha...			0x55555556037fc					
4	GThreadStd::run()			0x5555555e6616					
5	??			0x7ffff64dc2b3					
6	start_thread	pthread_create.c	442	0x7ffff6094b43					
7	clone3	clone3.S	81	0x7ffff6126a00					

Projects

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

Welcome

Edit

Design

Debug

Projects

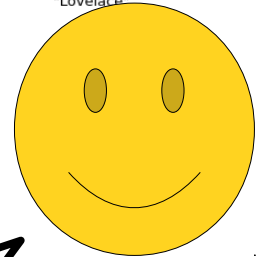
Help

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case
64      * lower
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71
72

```

Name	Value	Type
__for_begin	@0x7ffffb2ffc78	std::string::iter...
__for_end	@0x7ffffb2ffc80	std::string::iter...
__for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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



Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b							
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Debugger

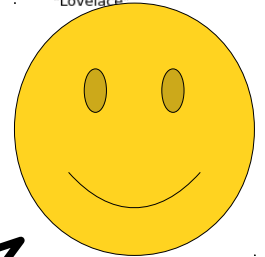
Debug

```

50  /* This hashing scheme needs two prime numbers, a large prime
51  * prime. These numbers were chosen because their product is
52  * 2^31 - kLargePrime - 1.
53  */
54  static const int kLargePrime = 15485863;
55  static const int kSmallPrime = 137;
56
57  int hashVal = 0;
58
59  /* Iterate across all the characters in the first name, then
60  * name, updating the hash at each step.
61  */
62  for (char ch: first + last) {
63      /* Convert the input character to lower case
64      * lower
65      */
66      ch = tolower(ch);
67      hashVal = (hashVal * kLargePrime + ch) % kSmallPrime;
68  }
69
70  return hashVal;
71
72

```

Name	Value	Type
for_begin	@0x7fffb2ffc78	std::string::iter...
for_end	@0x7fffb2ffc80	std::string::iter...
for_range	"AdaLovelace"	std::string &&
ch	'd'	char
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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



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

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555ab31b							
2	studentMain	NameHash.cpp	31	0x5555555ab0fb							
3	std::_Function_ha...			0x55555556037fc							
4	GThreadStd::run()			0x5555555e6616							
5	??			0x7ffff64dc2b3							
6	start_thread	pthread_create.c	442	0x7ffff6094b43							
7	clone3	clone3.S	81	0x7ffff6126a00							

NameHash [main]

NameHash.pro

Sources

NameHash.cpp

50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72

```

/* This hashing scheme needs two prime numbers, a large prime
 * prime. These numbers were chosen because their product is
 *  $2^{31} - kLargePrime - 1$ .
 */
static const int kLargePrime = 15485863;
static const int kSmallPrime = 137;

int hashVal = 0;

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

return hashVal;

```

Name	Value	Type
__for_begin	@0x7fffb2ffc78	std::string::iter...
__for_end	@0x7fffb2ffc80	std::string::iter...
__for_range	"Ada Lovelace"	std::string &&
ch	'd' 100	char 0x64
first	"Ada"	std::string
hashVal	?????	int
kLargePrime	15485863	int
kSmallPrime	137	int
last	"Lovelace"	std::string



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

NameHash

Debug

Debugger GDB for "NameHash" #15 NameHash Stopped: "end-stepping-range".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
1	nameHash	NameHash.cpp	62	0x5555555555555555	1	nameHash	NameHash.cpp	62	0x5555555555555555			
2	studentMain	NameHash.cpp	31	0x5555555555555555	2	studentMain	NameHash.cpp	31	0x5555555555555555			
3	std::_Function_ha...			0x5555555555555555	3	std::_Function_ha...			0x5555555555555555			
4	GThreadStd::run()			0x5555555555555555	4	GThreadStd::run()			0x5555555555555555			
5	??			0x7ffff64d00000000	5	??			0x7ffff64d00000000			
6	start_thread	pthread_create.c	442	0x7ffff6094b43	6	start_thread	pthread_create.c	442	0x7ffff6094b43			
7	clone3	clone3.S	81	0x7ffff6126a00	7	clone3	clone3.S	81	0x7ffff6126a00			

Projects

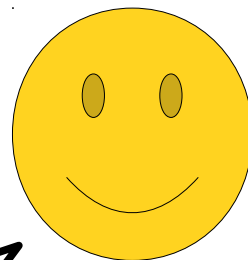
- NameHash [main]
 - NameHash.pro
 - Sources
 - NameHash.cpp

```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
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
38 * to talk mor
39 * the meantim
40 * of the input
41 *
42 * For those of you who are more mathematically inclined, this fi
43 * treats each character in the input name as a number between 0

```

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



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

Name	Value	Type
255	int	

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555ab0fb								
2	std::_Function_ha...			0x55555556037fc								
3	GThreadStd::run()			0x5555555e6616								
4	??			0x7ffff64dc2b3								
5	start_thread	pthread_create.c	442	0x7ffff6094b43								
6	clone3	clone3.S	81	0x7ffff6126a00								

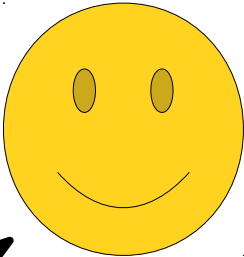
Projects NameHash.cpp main Unix (LF) Line: 31, Col: 5

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

```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
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
38 * to talk mor
39 * the meantim
40 * of the input
41 *
42 * For those of you who are more mathematically inclined, this fi
43 * treats each character in the input name as a number between 0

```



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

Name	Value	Type
255	int	

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555ab0fb							
2	std::_Function_ha...			0x55555556037fc							
3	GThreadStd::run()			0x5555555e6616							
4	??			0x7ffff64dc2b3							
5	start_thread	pthread_create.c	442	0x7ffff6094b43							
6	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

Edit

Design

Debug

Projects

Help

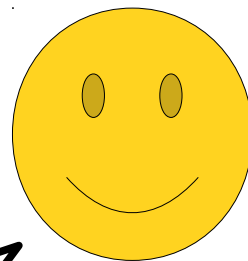
```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
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
38 * to talk mor
39 * the meantim
40 * of the input
41 *
42 * For those of you who are more mathematically inclined, this fi
43 * treats each character in the input name as a number between 0

```

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

Name	Value	Type
	255	int



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



Debugger GDB for "NameHash" Stopped: "function-finished".

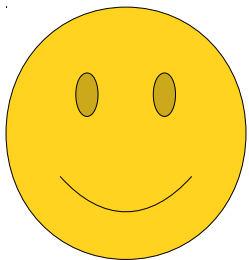
Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555ab0fb							
2	std::_Function_ha...			0x55555556037fc							
3	GThreadStd::run()			0x5555555e6616							
4	??			0x7ffff64dc2b3							
5	start_thread	pthread_create.c	442	0x7ffff6094b43							
6	clone3	clone3.S	81	0x7ffff6126a00							

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string first = getLine("What is your first name? ");
29     string last = getLine("What is your last name? ");
30
31     int hashValue = nameHash(first, last);
32
33     cout << "The hash of your name is: " << hashValue << endl;

```



We can see that the nameHash function returned 15058255. 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.)

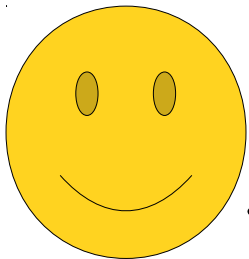
Name	Value	Type
first	"Ada"	std::string
hashValue	0	int
last	"Lovelace"	std::string

returned value	15058255	int
----------------	----------	-----

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

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string first = getLine("What is your first name? ");
29     string last = getLine("What is your last name? ");
30
31     int hashValue = nameHash(first, last);
32
33     cout << "The hash of your name is: " << hashValue << endl;
34     return 0;
35 }
36
```



But if we look up over here, we see that hashValue isn't storing 15058255, even though that's what was returned.

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

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

returned value 15058255 int

Name	Value	Type
0		

Views

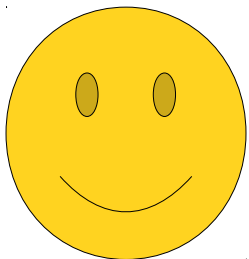
File Line Address Con/Ignore Threads

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

```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string first = getLine("What is your first name? ");
29     string last = getLine("What is your last name? ");
30
31     int hashValue = nameHash(first, last);
32
33     cout << "The hash of your name is: " << hashValue << endl;
34

```



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

returned value 15058255 int

We're
quarte
p the

this f
ween 0

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

Debugger GDB for "NameHash" #15 NameHash Stopped: "function-finished".

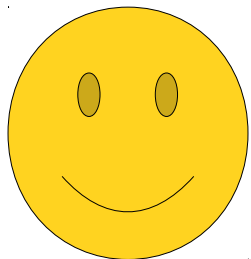
Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555ab0fb							
2	std::_Function_ha...			0x55555556037fc							
3	GThreadStd::run()			0x5555555e6616							
4	??			0x7ffff64dc2b3							
5	start_thread	pthread_create.c	442	0x7ffff6094b43							
6	clone3	clone3.S	81	0x7ffff6126a00							

NameHash [main]
NameHash.pro
Sources
NameHash.cpp

```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string first = getLine("What is your first name? ");
29     string last = getLine("What is your last name? ");
30
31     int hashValue = nameHash(first, last);
32
33     cout << "The hash of your name is: " << hashValue << endl;
34
35
36
37
38
39
40
41
42
43 * treats each character in the input name as a number between 0

```



This is pretty cool, actually!

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

returned value 15058255 int

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

Debugger GDB for "NameHash" Threads: #15 NameHash Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	studentMain	NameHash.cpp	31	0x5555555ab0fb							
2	std::_Function_ha...			0x55555556037fc							
3	GThreadStd::run()			0x5555555e6616							
4	??			0x7ffff64dc2b3							
5	start_thread	pthread_create.c	442	0x7ffff6094b43							
6	clone3	clone3.S	81	0x7ffff6126a00							

Projects

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

Welcome

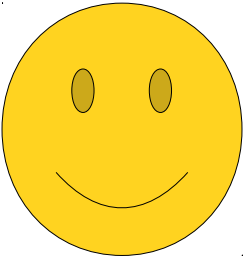
Edit

Design

Debug

Projects

Help



```

20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the func
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string first = getLine("What is your first name? ");
29     string last = getLine("What is your last name? ");
30
31     int hashValue = nameHash(first, last);
32
33     cout << "The hash of your name is: " << hashValue << endl;
34

```

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

returned value	15058255	int
----------------	----------	-----

Name	Value	Type

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!

Level	Function	File	Line	Address	Number	Function
1	studentMain	NameHash.cpp	31	0x5555555ab0fb		
2	std::_Function_ha...			0x55555556037fc		
3	GThreadStd::run()			0x5555555e6616		
4	??			0x7ffff64dc2b3		
5	start_thread	pthread_create.c	442	0x7ffff6094b43		
6	clone3	clone3.S	81	0x7ffff6126a00		

Projects

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

Welcome

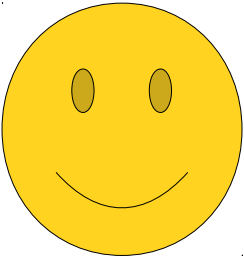
Edit

Design

Debug

Projects

Help



```

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

```

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

Name	Value	Type
returned value	15058255	int

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

Debugger

GDB for "NameHash" | Stopped: "function-finished".

Level	Function	File	Line	Address	Number	Function
1	studentMain	NameHash.cpp	31	0x55555ab0fb	1	studentMain
2	std::_Function_ha...			0x5555556037fc	2	std::_Function_ha...
3	GThreadStd::run()			0x555555e6616	3	GThreadStd::run()
4	??			0x55555564dc2b3	4	??
5	start_thread	pthread_create.c	442	0x7ffff6094b43	5	start_thread
6	clone3	clone3.S	81	0x7ffff6126a00	6	clone3

Projects

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

Welcome

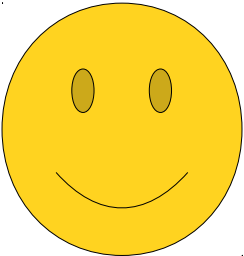
Edit

Design

Debug

Projects

Help



```

22  /* Prototype for the nameHash function. This lets us use the func
23  * in main and then define it later in the program.
24  */
25  int nameHash(string first, string last);
26
27  int main() {
28      string first = getLine("What is your first name? ");
29      string last = getLine("What is your last name? ");
30
31      int hashValue = nameHash(first, last);
32
33      cout << "The hash of your name is: " << hashValue << endl;
34      return 0;
35
36
44  * It then uses them as coefficients in a polynomial over the fi
45  * F_p, where p is a large prime number, and evaluates that poly

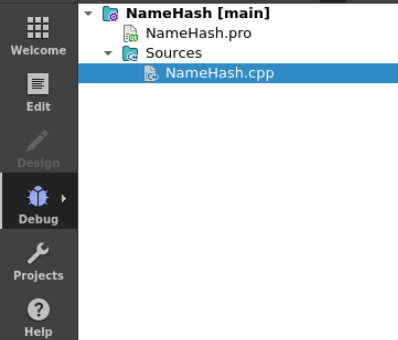
```

Name	Value	Type
first	"Ada"	std::string
hashValue	15058255	int
last	"Lovelace"	std::string

Name	Value	Type

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

Level	Function	File	Line	Address	Number	Function
1	studentMain	NameHash.cpp	33	0x5555555ab119		
2	std::_Function_ha...			0x55555556037fc		
3	GThreadStd::run()			0x5555555e6616		
4	??			0x7ffff64dc2b3		
5	start_thread	pthread_create.c	442	0x7ffff6094b43		
6	clone3	clone3.S	81	0x7ffff6126a00		

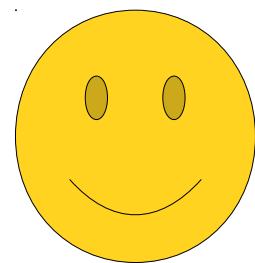


```

22  /* Prototype for the nameHash function. This lets us use the func
23  * in main and then define it later in the program.
24  */
25  int nameHash(string first, string last);
26
27  int main() {
28      string first = getLine("What is your first name? ");
29      string last = getLine("What is your last name? ");
30
31      int hashValue = nameHash(first, last);
32
33      cout << "The hash of your name is: " << hashValue << endl;
34      return 0;
35  }
36
37  /* This is the actual function that computes the hash and
38  * to talk
39  * the
40  * of
41  *
42  * For
43  * tre
44  * It
45  * F_p

```

Name	Value	Type
first	"Ada"	std::string
hashValue	15058255	int
last	"Lovelace"	std::string



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



Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con/Ignore	Threads
1	studentMain	NameHash.cpp	33	0x5555555ab119							
2	std::Function_ha...			0x55555556037fc							
3	GThreadStd::run()			0x5555555e6616							
4	??			0x7ffff64dc2b3							
5	start_thread	pthread_create.c	442	0x7ffff6094b43							
6	clone3	clone3.S	81	0x7ffff6126a00							

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

```

22  /* Prototype for the nameHash function. This lets us use the func
23  * in main and then define it later in the program.
24  */
25  int nameHash(string first, string last);
26
27  int main() {
28
29
30
31  What is your first name? Ada
32  What is your last name? Lovelace
33  The hash of your name is: 15058255
34
35
36
37
38
39
40
41
42
43
44  * It th
45  * F_p, where p is a large prime number, and evaluates that poly

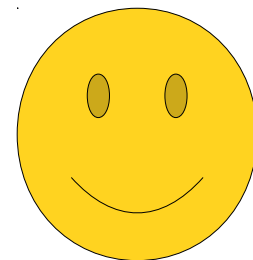
```

NameHash Console [Completed]

```

File Edit Options Help
What is your first name? Ada
What is your last name? Lovelace
The hash of your name is: 15058255

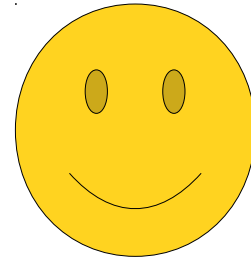
```



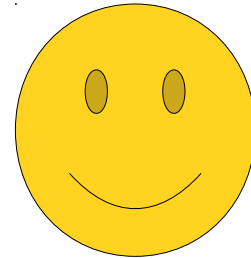
If you do, you should see something like this.
 (The program window might not automatically
 pop up. That's okay! Just open it manually.)
 Our program is now done running!

Level	Function	File	Line	Address	Number	Function	File	Line	Address	Con	Ignore	Threads
1	studentMain	NameHash.cpp	33	0x5555555ab119								
2	std::function_ha...			0x55555556037fc								
3	GThreadStd::run()			0x5555555e6616								
4	??			0x7ffff64dc2b3								
5	start_thread	pthread_create.c	442	0x7ffff6094b43								
6	clone3	clone3.S	81	0x7ffff6126a00								

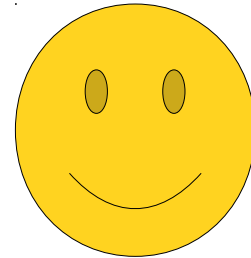
So there you have it! You've now gotten more familiar with the debugger!



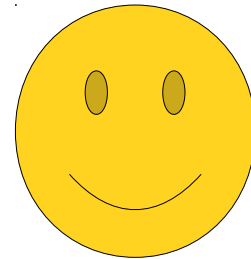
You know how to set a breakpoint to pause the program at a particular point.



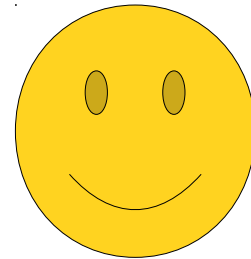
You know how to read the call stack and to see the values of local variables.



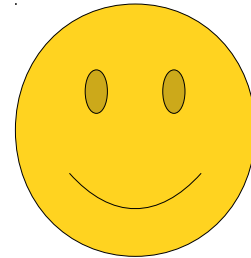
You know how to single-step the program and see what values change.



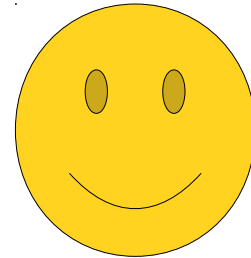
You know how to run a function to completion,
and how to let the program keep on running.



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!

