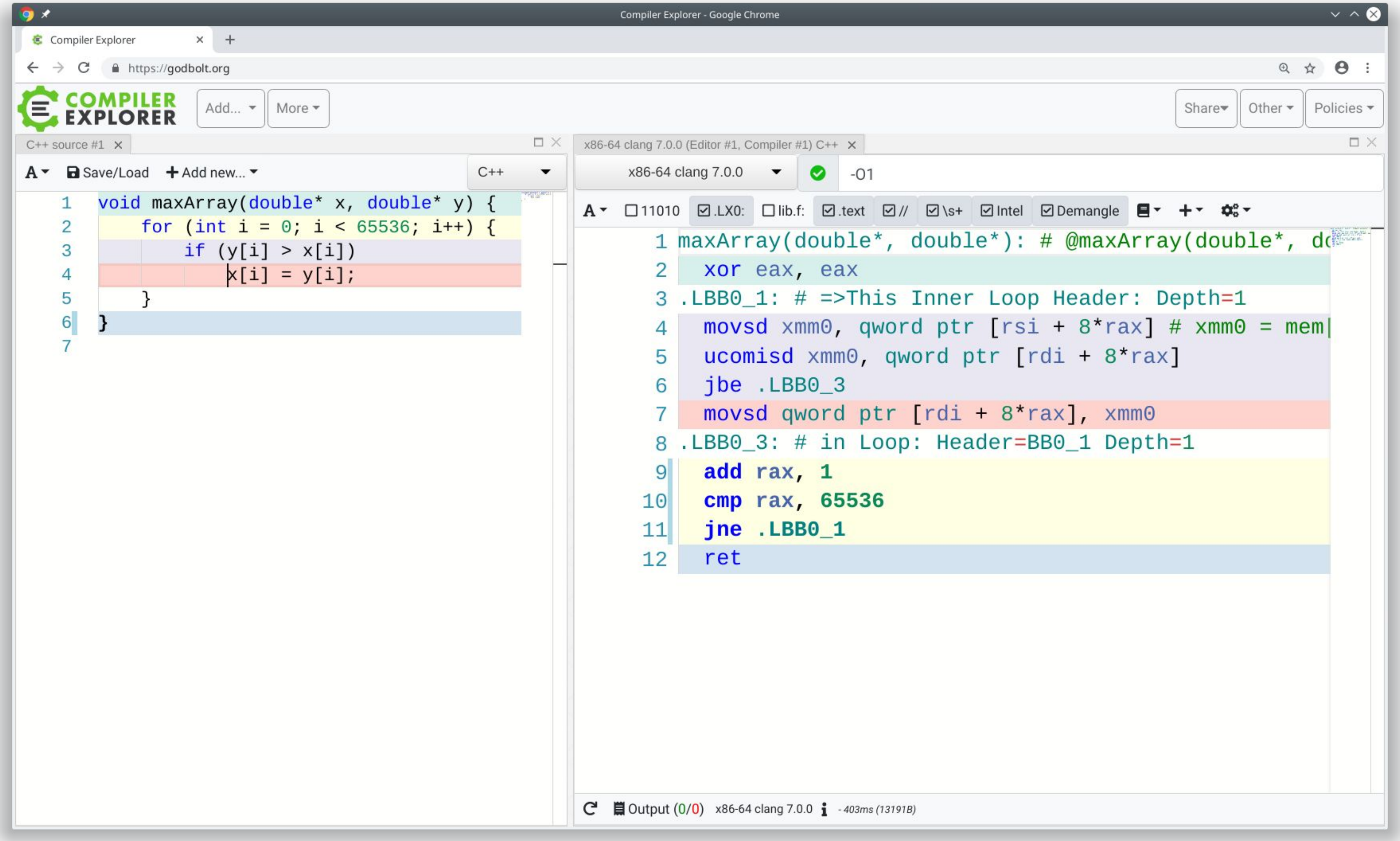
COMPILER EXPLORER

https://godbolt.org by Matt Godbolt

Code analysieren und Diskussionen gewinnen

Was ist das?



```cpp
void maxArray(double* x, double* y) {
    for (int i = 0; i < 65536; i++) {
        if (y[i] > x[i])
            x[i] = y[i];
    }
}
```

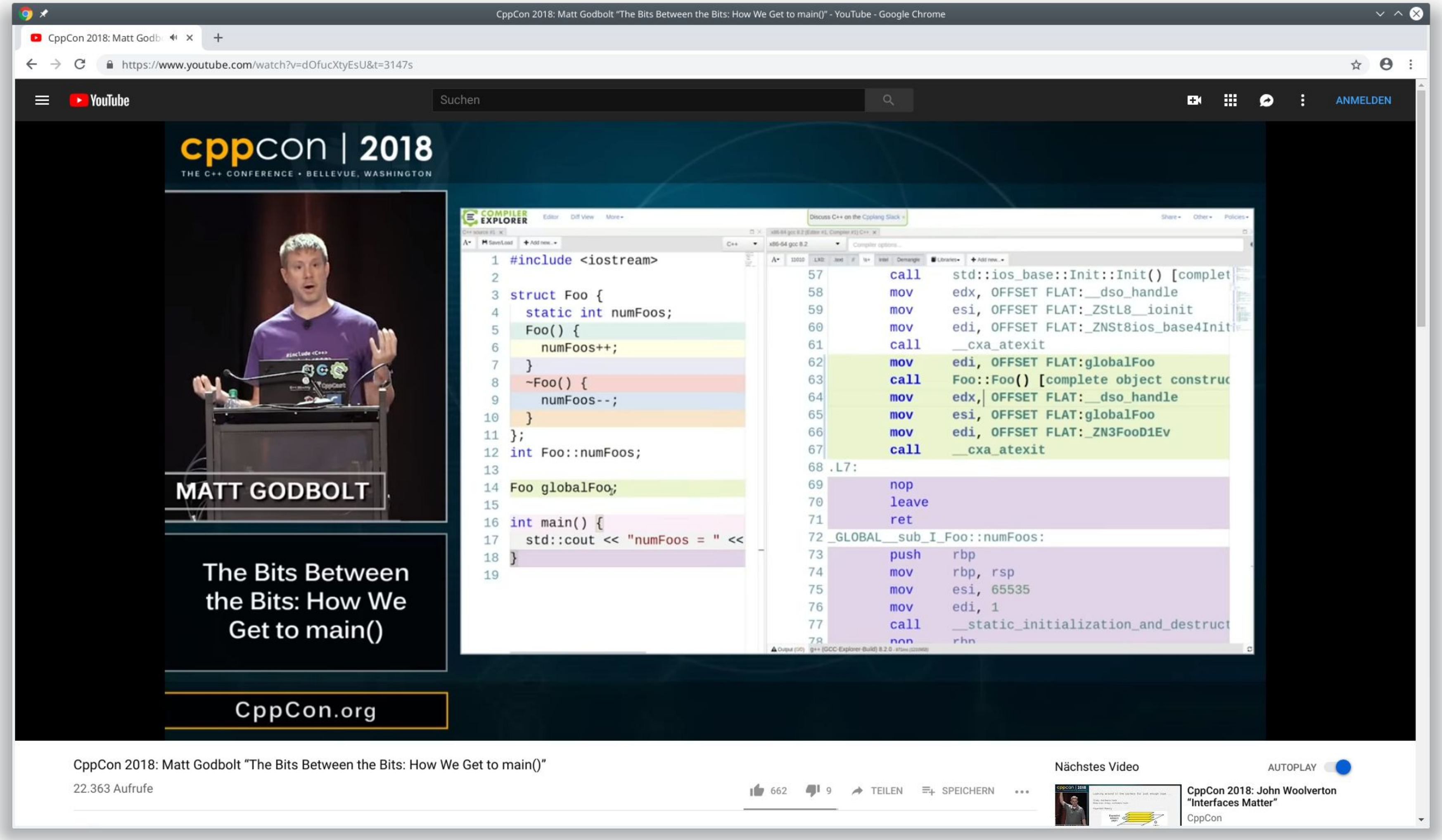
```
maxArray(double*, double*): # @maxArray(double*, dc
  xor eax, eax
.LBB0_1: # =>This Inner Loop Header: Depth=1
  movsd xmm0, qword ptr [rsi + 8*rax] # xmm0 = mem
  ucomisd xmm0, qword ptr [rdi + 8*rax]
  jbe .LBB0_3
  movsd qword ptr [rdi + 8*rax], xmm0
.LBB0_3: # in Loop: Header=BB0_1 Depth=1
  add rax, 1
  cmp rax, 65536
  jne .LBB0_1
  ret
```

https://godbolt.org/z/-MpVCV

Was ist das?

- Online Tool von Matt Godbolt
- Compiler Optimierungen aufzeigen
- Code schreiben, Assembler vergleichen
- Compiler und neue Features testen
- Code teilen und Wissen erweitern

“The Bits Between the Bits: How We Get to main()”



CppCon 2018: Matt Godbolt “The Bits Between the Bits: How We Get to main()” - YouTube - Google Chrome

https://www.youtube.com/watch?v=dOfucXtyEsU&t=3147s

YouTube
Suchen
ANMELDEN
cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON
MATT GODBOLT
The Bits Between the Bits: How We Get to main()
CppCon.org
COMPILER EXPLORER

```
#include <iostream>

struct Foo {
  static int numFoos;
  Foo() {
    numFoos++;
  }
  ~Foo() {
    numFoos--;
  }
};
int Foo::numFoos;

Foo globalFoo;

int main() {
  std::cout << "numFoos = " <<
}
```

```
        call    std::ios_base::Init::Init() [complet
        mov     edx, OFFSET FLAT:__dso_handle
        mov     esi, OFFSET FLAT:_ZStL8__ioinit
        mov     edi, OFFSET FLAT:_ZNSt8ios_base4Init
        call    __cxa_atexit
        mov     edi, OFFSET FLAT:globalFoo
        call    Foo::Foo() [complete object construc
        mov     edx, OFFSET FLAT:__dso_handle
        mov     esi, OFFSET FLAT:globalFoo
        mov     edi, OFFSET FLAT:_ZN3FooD1Ev
        call    __cxa_atexit
.L7:
        nop
        leave
        ret
_GLOBAL__sub_I_Foo::numFoos:
        push    rbp
        mov     rbp, rsp
        mov     esi, 65535
        mov     edi, 1
        call    __static_initialization_and_destruct
        pop     rbp
```

CppCon 2018: Matt Godbolt “The Bits Between the Bits: How We Get to main()”

22.363 Aufrufe

662 9 TEILEN SPEICHERN

Nächstes Video AUTOPLAY

CppCon 2018: John Woolverton “Interfaces Matter”

CppCon

https://www.youtube.com/watch?v=dOfucXtyEsU

Wofür brauche ich das?

- Compiler verstehen und vergleichen
- Neue Compiler Features demonstrieren
  - z.B. auf der CppCon genutzt
- Sprachen vergleichen (Rust <--> C++)
- Argumente gewinnen
  “Memset ist schneller”, “STL ist langsam”

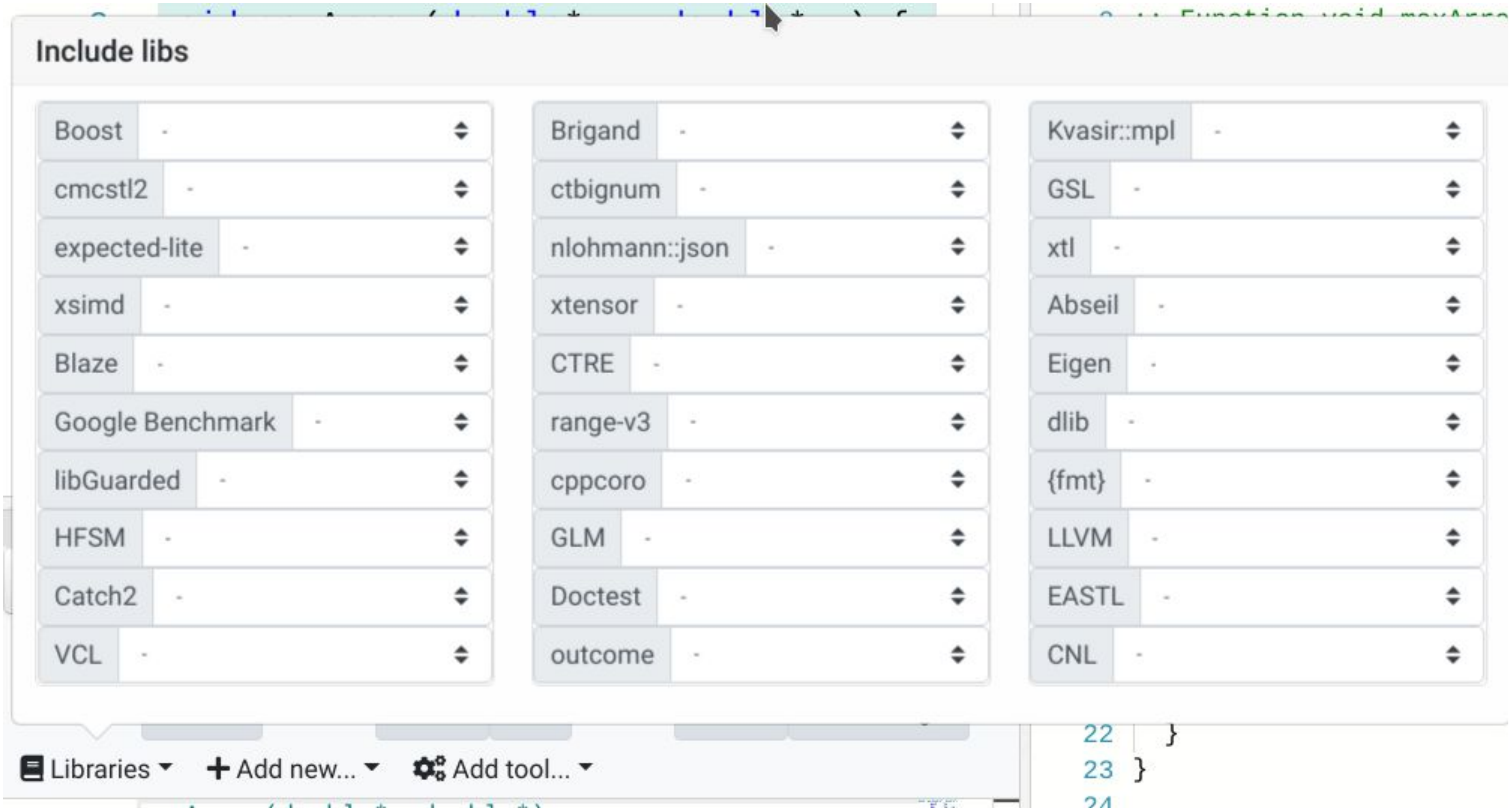
Welche Sprachen gibt es auf godbolt.org?

- C
- C++
- Rust
- D
- Go
- Haskell
- Swift
- Pascal
- Fortran
- Assembler
- ZIG
- ...

C++ Compiler

- GCC 4.1, 4.4, 4.5, ..., 6, ..., 7.1, ..., 8, trunk
  - ARM GCC, MIPS GCC, MSP GCC
- Clang 3.0, 3.1, ..., 4, 5, 6, 7, trunk, Cppx (-Wlifetime)
- ICC 13, 16, 17, 18, 19
- MSVC 19 (unter Windows und Wine)
- Compiler für PowerPC
- ...

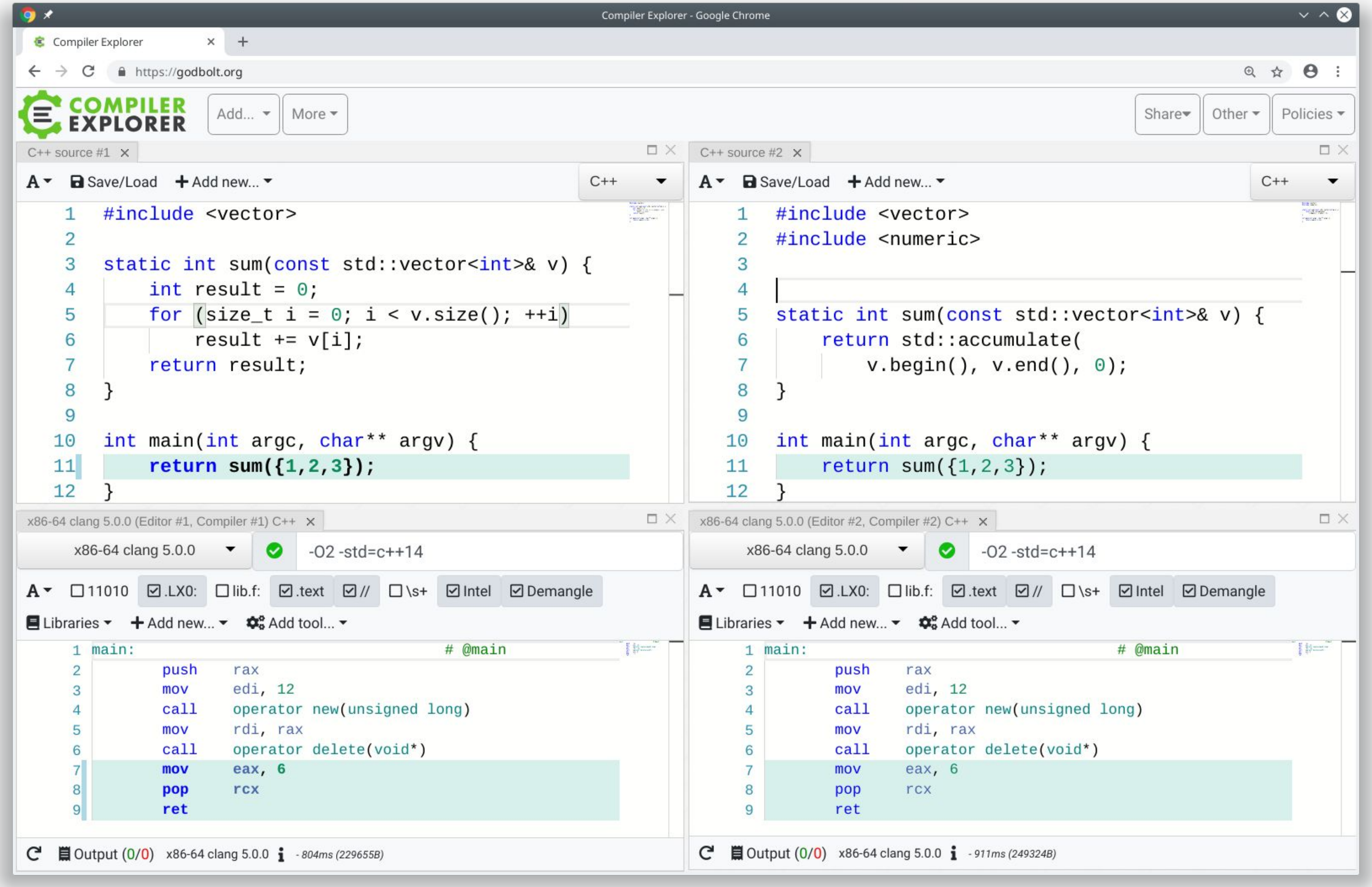
C++ Bibliotheken



Include libs

Boost -	Brigand -	Kvasir::mpl -
cmcstl2 -	ctbignum -	GSL -
expected-lite -	nlohmann::json -	xtl -
xsimd -	xtensor -	Abseil -
Blaze -	CTRE -	Eigen -
Google Benchmark -	range-v3 -	dlib -
libGuarded -	cppcoro -	{fmt} -
HFSM -	GLM -	LLVM -
Catch2 -	Doctest -	EASTL -
VCL -	outcome -	CNL -

Features | Mehrere Editoren


Compiler Explorer - Google Chrome
https://godbolt.org
Add... More Share Other Policies
C++ source #1
Save/Load Add new... C++
C++ source #2
x86-64 clang 5.0.0 (Editor #1, Compiler #1) C++
x86-64 clang 5.0.0 (Editor #2, Compiler #2) C++
11010 .LX0: lib.f: .text // \s+ Intel Demangle
Libraries Add new... Add tool...
Output (0/0) x86-64 clang 5.0.0 - 804ms (229655B)
Output (0/0) x86-64 clang 5.0.0 - 911ms (249324B)

```cpp
#include <vector>

static int sum(const std::vector<int>& v) {
    int result = 0;
    for (size_t i = 0; i < v.size(); ++i)
        result += v[i];
    return result;
}

int main(int argc, char** argv) {
    return sum({1,2,3});
}
```

```cpp
#include <vector>
#include <numeric>


static int sum(const std::vector<int>& v) {
    return std::accumulate(
        v.begin(), v.end(), 0);
}

int main(int argc, char** argv) {
    return sum({1,2,3});
}
```

x86-64 clang 5.0.0 `-O2 -std=c++14`

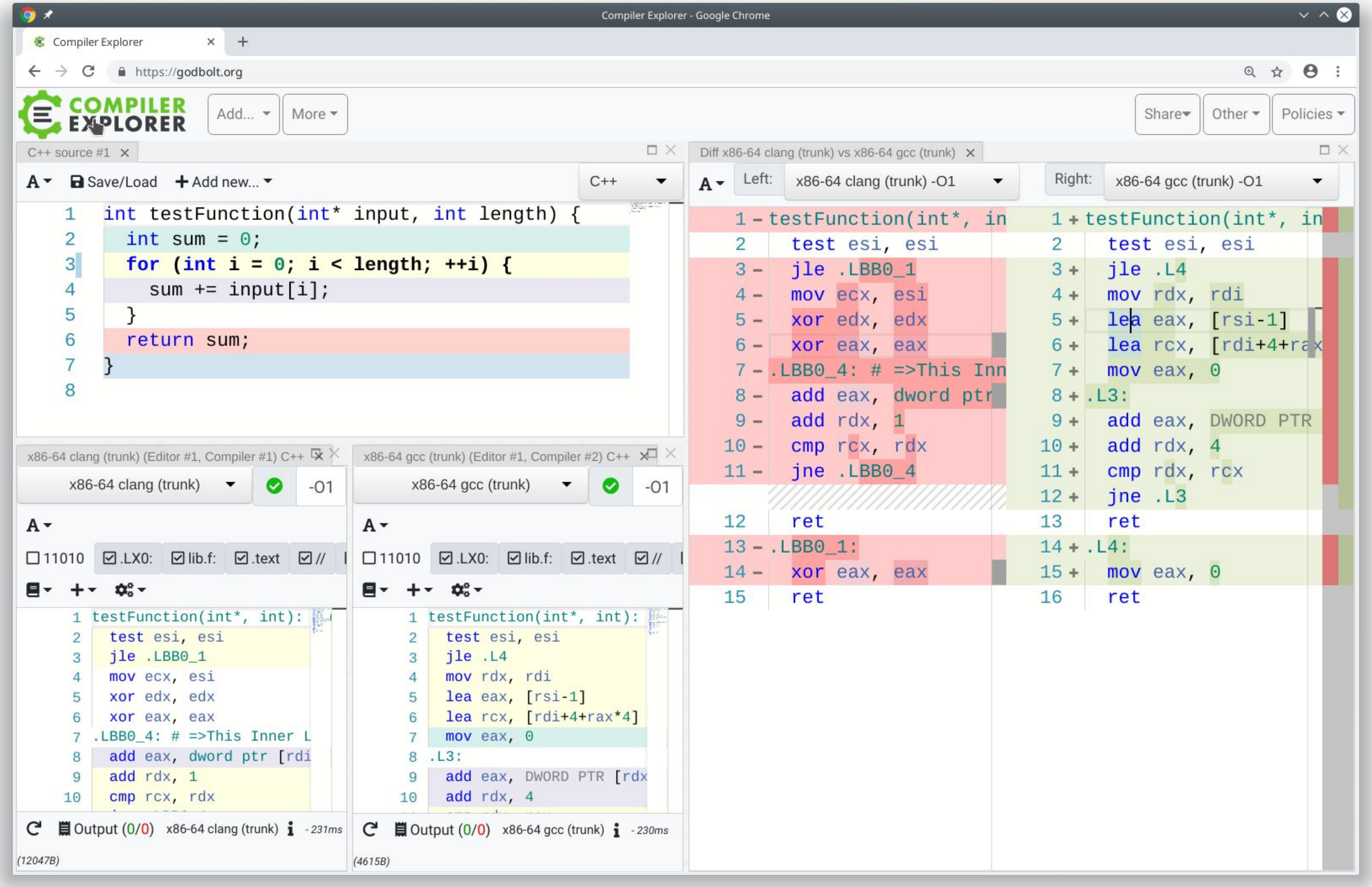
```
main:                                   # @main
        push    rax
        mov     edi, 12
        call    operator new(unsigned long)
        mov     rdi, rax
        call    operator delete(void*)
        mov     eax, 6
        pop     rcx
        ret
```

x86-64 clang 5.0.0 `-O2 -std=c++14`

```
main:                                   # @main
        push    rax
        mov     edi, 12
        call    operator new(unsigned long)
        mov     rdi, rax
        call    operator delete(void*)
        mov     eax, 6
        pop     rcx
        ret
```

https://godbolt.org/z/IqZ4LJ

Features | Diff View



C++ source #1

```
int testFunction(int* input, int length) {
  int sum = 0;
  for (int i = 0; i < length; ++i) {
    sum += input[i];
  }
  return sum;
}
```

x86-64 clang (trunk) (Editor #1, Compiler #1) C++ — x86-64 clang (trunk) -O1

```
testFunction(int*, int):
  test esi, esi
  jle .LBB0_1
  mov ecx, esi
  xor edx, edx
  xor eax, eax
.LBB0_4: # =>This Inner L
  add eax, dword ptr [rdi
  add rdx, 1
  cmp rcx, rdx
```

Output (0/0) x86-64 clang (trunk) - 231ms (12047B)

x86-64 gcc (trunk) (Editor #1, Compiler #2) C++ — x86-64 gcc (trunk) -O1

```
testFunction(int*, int):
  test esi, esi
  jle .L4
  mov rdx, rdi
  lea eax, [rsi-1]
  lea rcx, [rdi+4+rax*4]
  mov eax, 0
.L3:
  add eax, DWORD PTR [rdx
  add rdx, 4
```

Output (0/0) x86-64 gcc (trunk) - 230ms (4615B)

Diff x86-64 clang (trunk) vs x86-64 gcc (trunk)

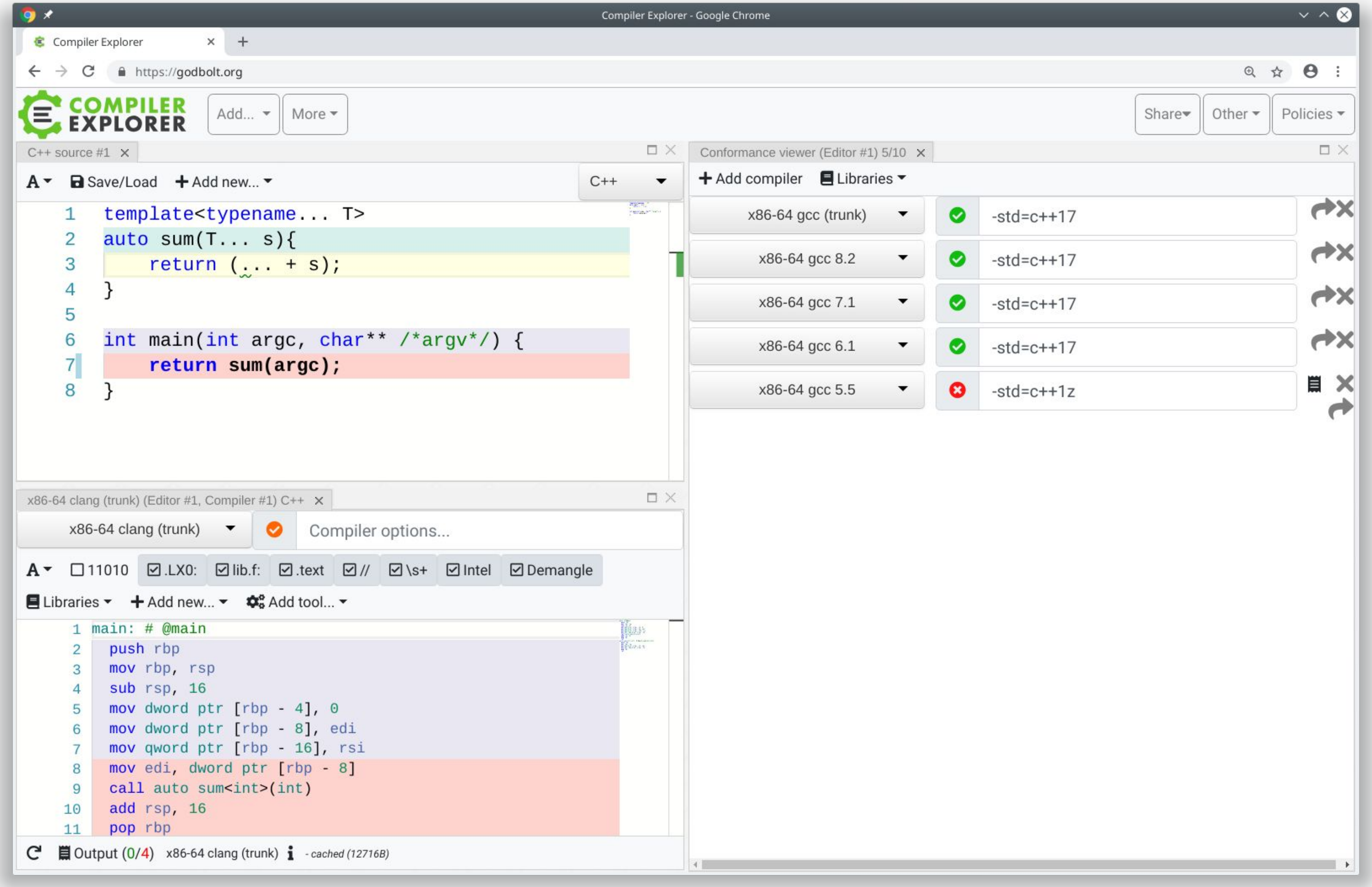
Left: x86-64 clang (trunk) -O1

```
- testFunction(int*, in
    test esi, esi
-   jle .LBB0_1
-   mov ecx, esi
-   xor edx, edx
-   xor eax, eax
- .LBB0_4: # =>This Inn
-   add eax, dword ptr
-   add rdx, 1
-   cmp rcx, rdx
-   jne .LBB0_4

    ret
- .LBB0_1:
-   xor eax, eax
    ret
```

Right: x86-64 gcc (trunk) -O1

```
+ testFunction(int*, in
    test esi, esi
+   jle .L4
+   mov rdx, rdi
+   lea eax, [rsi-1]
+   lea rcx, [rdi+4+rax
+   mov eax, 0
+ .L3:
+   add eax, DWORD PTR
+   add rdx, 4
+   cmp rdx, rcx
+   jne .L3
    ret
+ .L4:
+   mov eax, 0
    ret
```

Features | Conformance View

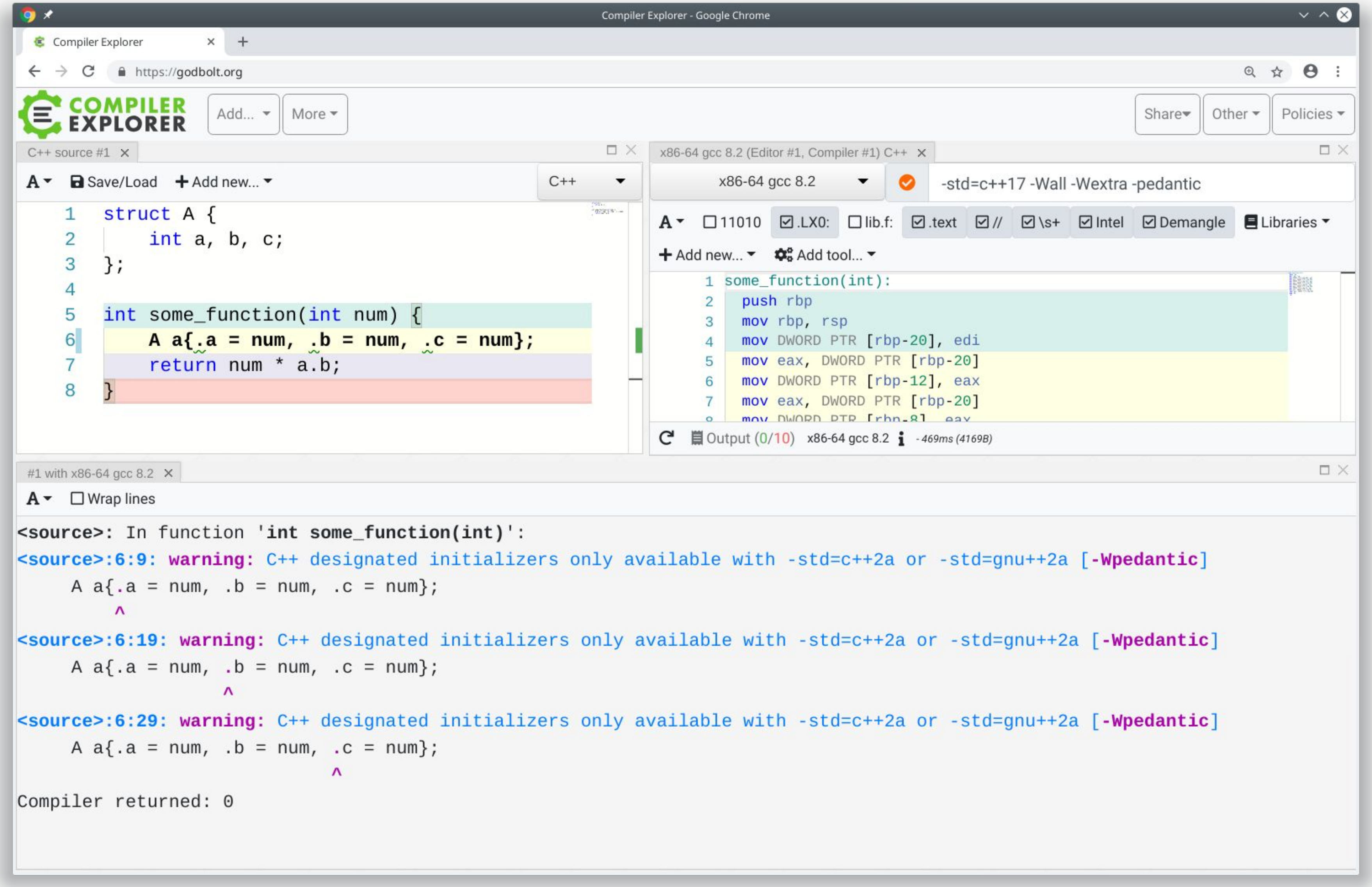


```cpp
template<typename... T>
auto sum(T... s){
    return (... + s);
}

int main(int argc, char** /*argv*/) {
    return sum(argc);
}
```

x86-64 gcc (trunk)	✓	-std=c++17
x86-64 gcc 8.2	✓	-std=c++17
x86-64 gcc 7.1	✓	-std=c++17
x86-64 gcc 6.1	✓	-std=c++17
x86-64 gcc 5.5	✗	-std=c++1z

```asm
main: # @main
  push rbp
  mov rbp, rsp
  sub rsp, 16
  mov dword ptr [rbp - 4], 0
  mov dword ptr [rbp - 8], edi
  mov qword ptr [rbp - 16], rsi
  mov edi, dword ptr [rbp - 8]
  call auto sum<int>(int)
  add rsp, 16
  pop rbp
```

Features | Konsole



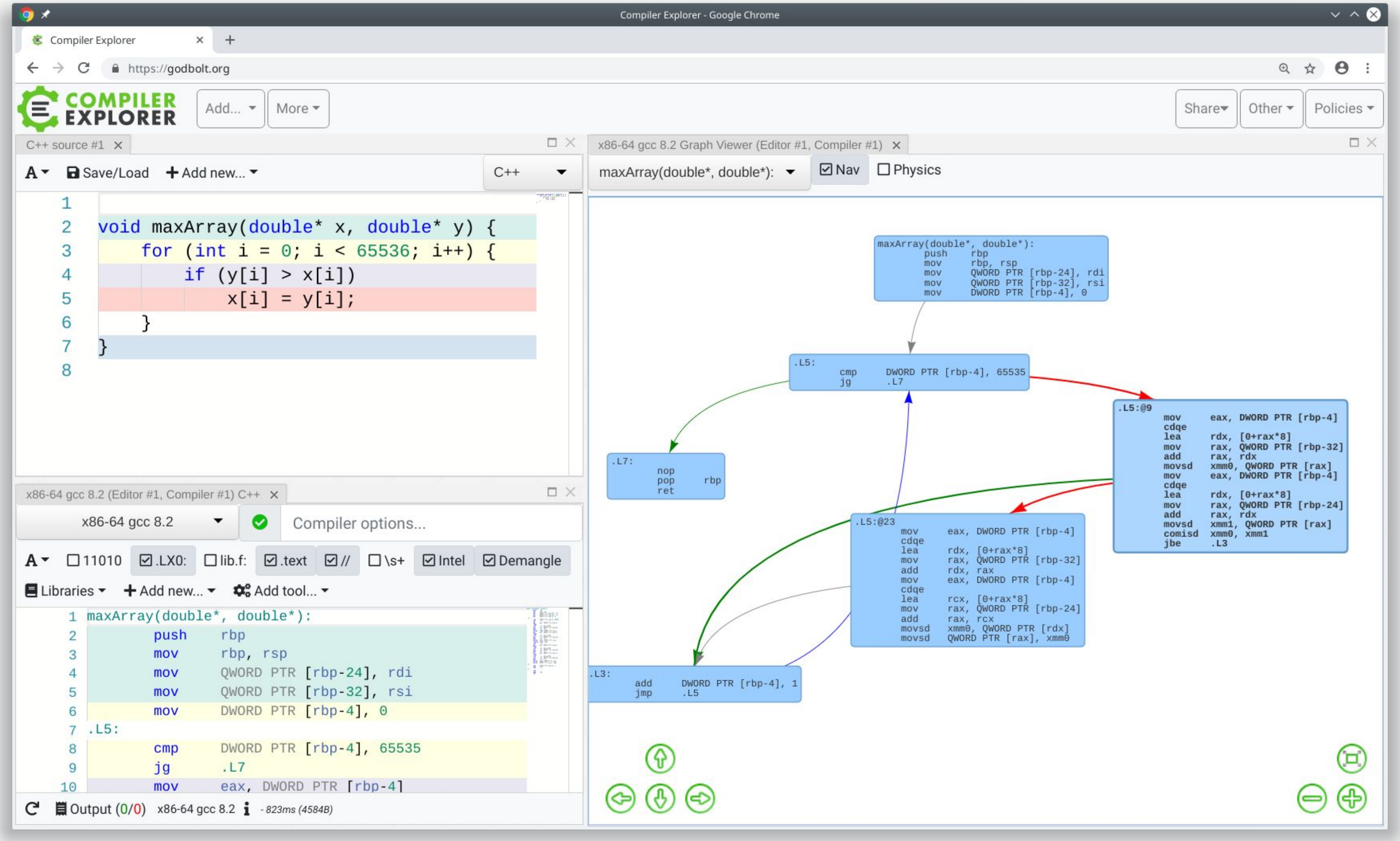
Compiler options: -std=c++17 -Wall -Wextra -pedantic

```cpp
struct A {
    int a, b, c;
};

int some_function(int num) {
    A a{.a = num, .b = num, .c = num};
    return num * a.b;
}
```

```
some_function(int):
  push rbp
  mov rbp, rsp
  mov DWORD PTR [rbp-20], edi
  mov eax, DWORD PTR [rbp-20]
  mov DWORD PTR [rbp-12], eax
  mov eax, DWORD PTR [rbp-20]
  mov DWORD PTR [rbp-8], eax
```

```
<source>: In function 'int some_function(int)':
<source>:6:9: warning: C++ designated initializers only available with -std=c++2a or -std=gnu++2a [-Wpedantic]
     A a{.a = num, .b = num, .c = num};
         ^
<source>:6:19: warning: C++ designated initializers only available with -std=c++2a or -std=gnu++2a [-Wpedantic]
     A a{.a = num, .b = num, .c = num};
                   ^
<source>:6:29: warning: C++ designated initializers only available with -std=c++2a or -std=gnu++2a [-Wpedantic]
     A a{.a = num, .b = num, .c = num};
                             ^
Compiler returned: 0
```

Features | Tools: GCC Graph Viewer



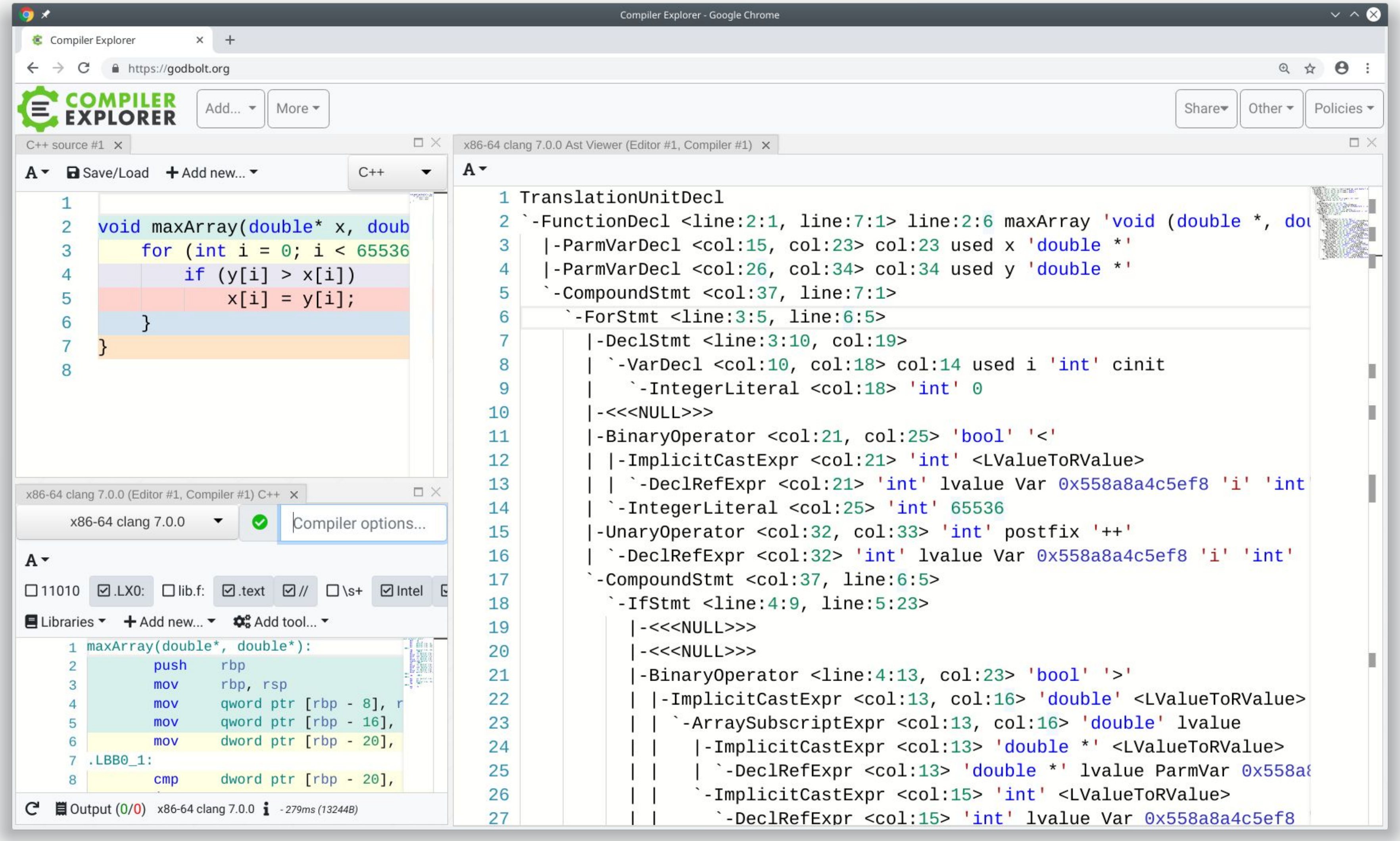
```cpp

void maxArray(double* x, double* y) {
    for (int i = 0; i < 65536; i++) {
        if (y[i] > x[i])
            x[i] = y[i];
    }
}

```

```
maxArray(double*, double*):
        push    rbp
        mov     rbp, rsp
        mov     QWORD PTR [rbp-24], rdi
        mov     QWORD PTR [rbp-32], rsi
        mov     DWORD PTR [rbp-4], 0
.L5:
        cmp     DWORD PTR [rbp-4], 65535
        jg      .L7
        mov     eax, DWORD PTR [rbp-4]
```

```
maxArray(double*, double*):
        push    rbp
        mov     rbp, rsp
        mov     QWORD PTR [rbp-24], rdi
        mov     QWORD PTR [rbp-32], rsi
        mov     DWORD PTR [rbp-4], 0

.L5:
        cmp     DWORD PTR [rbp-4], 65535
        jg      .L7

.L5:@9
        mov     eax, DWORD PTR [rbp-4]
        cdqe
        lea     rdx, [0+rax*8]
        mov     rax, QWORD PTR [rbp-32]
        add     rax, rdx
        movsd   xmm0, QWORD PTR [rax]
        mov     eax, DWORD PTR [rbp-4]
        cdqe
        lea     rdx, [0+rax*8]
        mov     rax, QWORD PTR [rbp-24]
        add     rax, rdx
        movsd   xmm1, QWORD PTR [rax]
        comisd  xmm0, xmm1
        jbe     .L3

.L7:
        nop
        pop     rbp
        ret

.L5:@23
        mov     eax, DWORD PTR [rbp-4]
        cdqe
        lea     rdx, [0+rax*8]
        mov     rax, QWORD PTR [rbp-32]
        add     rdx, rax
        mov     eax, DWORD PTR [rbp-4]
        cdqe
        lea     rcx, [0+rax*8]
        mov     rax, QWORD PTR [rbp-24]
        add     rax, rcx
        movsd   xmm0, QWORD PTR [rdx]
        movsd   QWORD PTR [rax], xmm0

.L3:
        add     DWORD PTR [rbp-4], 1
        jmp     .L5
```

https://godbolt.org/z/02h5V7

Features | Tools: Clang AST



```
void maxArray(double* x, doub
    for (int i = 0; i < 65536
        if (y[i] > x[i])
            x[i] = y[i];
    }
}
```

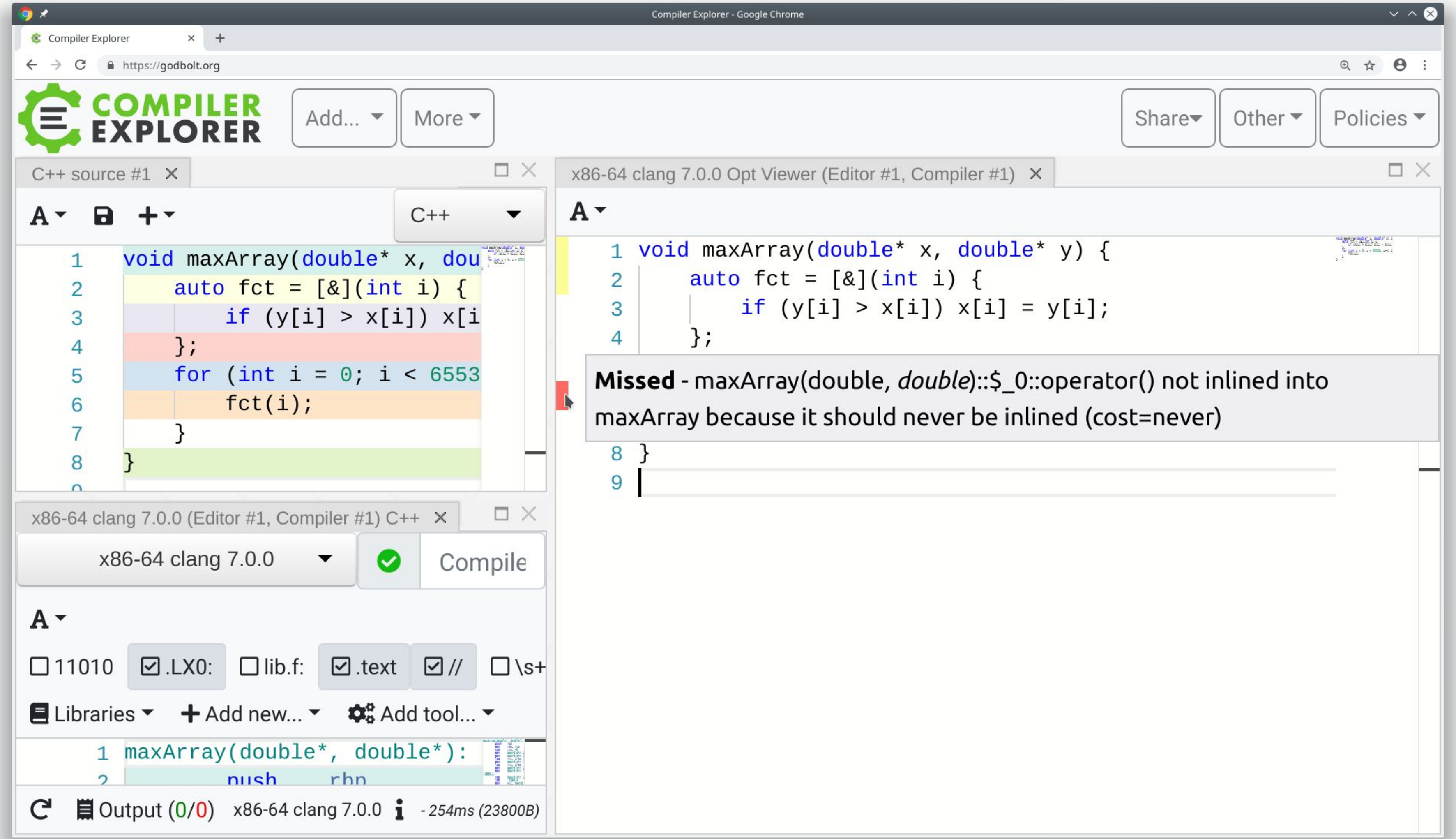
☐ 11010 ☑ .LX0: ☐ lib.f: ☑ .text ☑ // ☐ \s+ ☑ Intel

```
maxArray(double*, double*):
        push    rbp
        mov     rbp, rsp
        mov     qword ptr [rbp - 8], r
        mov     qword ptr [rbp - 16],
        mov     dword ptr [rbp - 20],
.LBB0_1:
        cmp     dword ptr [rbp - 20],
```

Output (0/0) x86-64 clang 7.0.0 - 279ms (13244B)

```
TranslationUnitDecl
`-FunctionDecl <line:2:1, line:7:1> line:2:6 maxArray 'void (double *, dou
  |-ParmVarDecl <col:15, col:23> col:23 used x 'double *'
  |-ParmVarDecl <col:26, col:34> col:34 used y 'double *'
  `-CompoundStmt <col:37, line:7:1>
    `-ForStmt <line:3:5, line:6:5>
      |-DeclStmt <line:3:10, col:19>
      | `-VarDecl <col:10, col:18> col:14 used i 'int' cinit
      |   `-IntegerLiteral <col:18> 'int' 0
      |-<<<NULL>>>
      |-BinaryOperator <col:21, col:25> 'bool' '<'
      | |-ImplicitCastExpr <col:21> 'int' <LValueToRValue>
      | | `-DeclRefExpr <col:21> 'int' lvalue Var 0x558a8a4c5ef8 'i' 'int'
      | `-IntegerLiteral <col:25> 'int' 65536
      |-UnaryOperator <col:32, col:33> 'int' postfix '++'
      | `-DeclRefExpr <col:32> 'int' lvalue Var 0x558a8a4c5ef8 'i' 'int'
      `-CompoundStmt <col:37, line:6:5>
        `-IfStmt <line:4:9, line:5:23>
          |-<<<NULL>>>
          |-<<<NULL>>>
          |-BinaryOperator <line:4:13, col:23> 'bool' '>'
          | |-ImplicitCastExpr <col:13, col:16> 'double' <LValueToRValue>
          | | `-ArraySubscriptExpr <col:13, col:16> 'double' lvalue
          | |   |-ImplicitCastExpr <col:13> 'double *' <LValueToRValue>
          | |   | `-DeclRefExpr <col:13> 'double *' lvalue ParmVar 0x558a8
          | |   `-ImplicitCastExpr <col:15> 'int' <LValueToRValue>
          | |     `-DeclRefExpr <col:15> 'int' lvalue Var 0x558a8a4c5ef8
```

Features | Opt View



```
void maxArray(double* x, dou
    auto fct = [&](int i) {
        if (y[i] > x[i]) x[i
    };
    for (int i = 0; i < 6553
        fct(i);
    }
}
```

```
void maxArray(double* x, double* y) {
    auto fct = [&](int i) {
        if (y[i] > x[i]) x[i] = y[i];
    };
```

Missed - maxArray(double, *double*)::$_0::operator() not inlined into maxArray because it should never be inlined (cost=never)

```
}
```

```
maxArray(double*, double*):
        push    rbp
```

Wie geht’s weiter?

- mit godbolt.org spielen und austauschen
- lokale Instanz installieren
- andere Sprachen testen