Reverse Engineering

2023/5/7, Sharkkcode

Sharkkcode

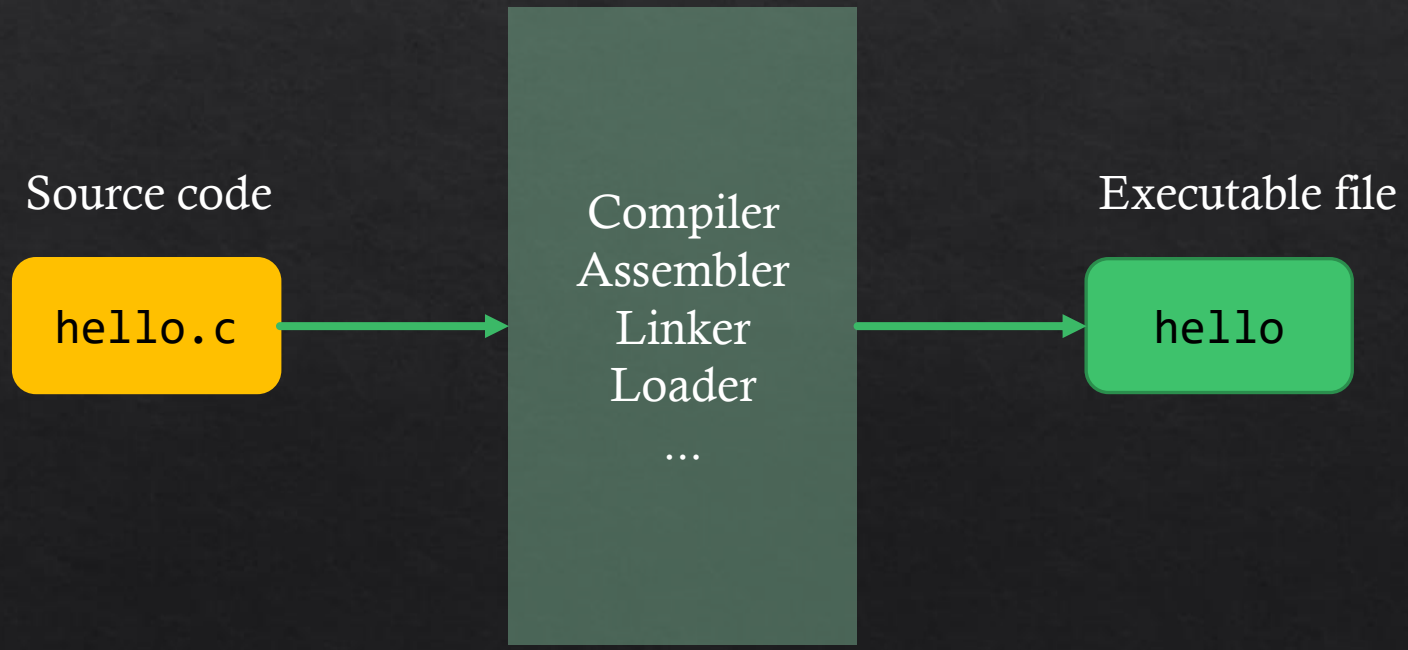
- CCU IM → NTHU IS
- CCU ISC
- Interested in Reverse, PWN
- BLOG :
  - https://sharkkcode.github.io/

Outline

- Reverse Engineering Overview
- Analysis
- x64 Assembly
- Memory Layout
- Demo Learning Website
- Lab
- Tools
  - GHIDRA
  - GDB
  - pwntools
- Lab

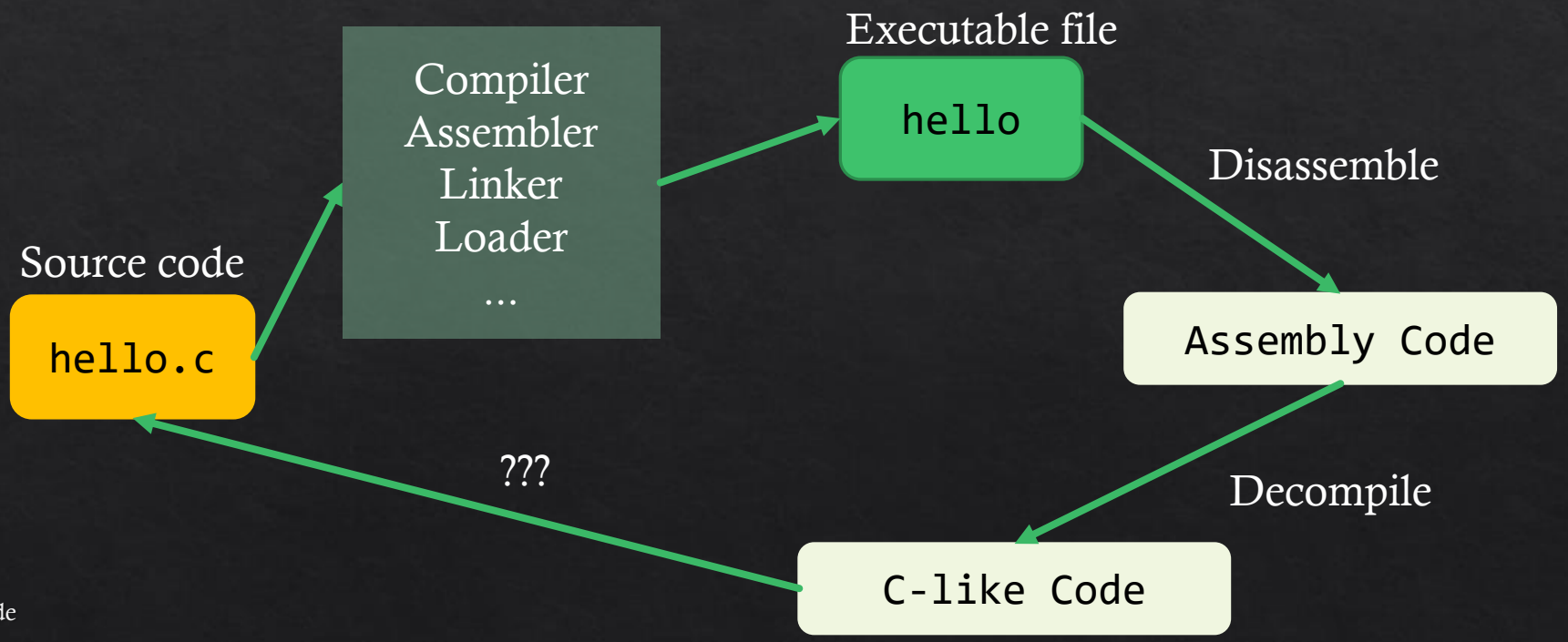
Reverse Engineering Overview

- When we finish writing a program `hello.c` , we might want to use Compiler, Assembler, Linker, Loader to generate an executable file `hello` to execute.



Reverse Engineering Overview

- So what exactly is Reverse Engineering doing?
  - Analyzing the files and information you have to reconstruct the behavior of a program or project, which can facilitate further actions such as writing a cracking program, etc.



Bigger Overview?

- Figure out **what is going on** …

通靈

Analysis

- Statically
  - Without running…
- Dynamically
  - When running…

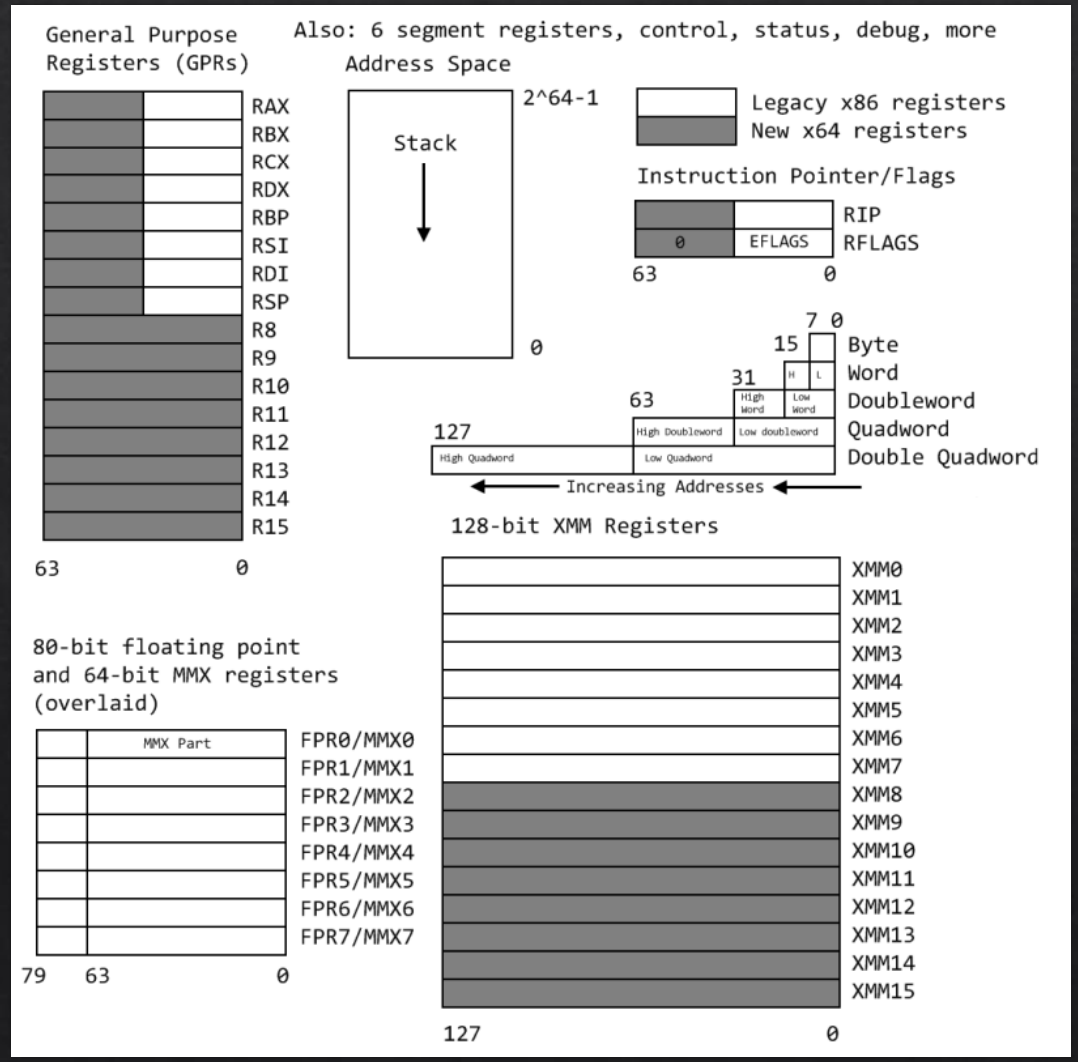
x64 Assembly

- https://www.intel.com/content/dam/develop/external/us/en/documents/introduction-to-x64-assembly-181178.pdf

x64 Assembly -- Cheatsheet

- https://cs.brown.edu/courses/cs033/docs/guides/x64_cheatsheet.pdf

x64 Assembly -- General Architecture



x64 Assembly -- Registers

4.3 Register Usage

There are sixteen 64-bit registers in x86-64: **%rax**, **%rbx**, **%rcx**, **%rdx**, **%rdi**, **%rsi**, **%rbp**, **%rsp**, and **%r8-r15**. Of these, **%rax**, **%rcx**, **%rdx**, **%rdi**, **%rsi**, **%rsp**, and **%r8-r11** are considered caller-save registers, meaning that they are not necessarily saved across function calls. By convention, **%rax** is used to store a function's return value, if it exists and is no more than 64 bits long. (Larger return types like structs are returned using the stack.) Registers **%rbx**, **%rbp**, and **%r12-r15** are callee-save registers, meaning that they are saved across function calls. Register **%rsp** is used as the *stack pointer*, a pointer to the topmost element in the stack.

Additionally, **%rdi**, **%rsi**, **%rdx**, **%rcx**, **%r8**, and **%r9** are used to pass the first six integer or pointer parameters to called functions. Additional parameters (or large parameters such as structs passed by value) are passed on the stack.

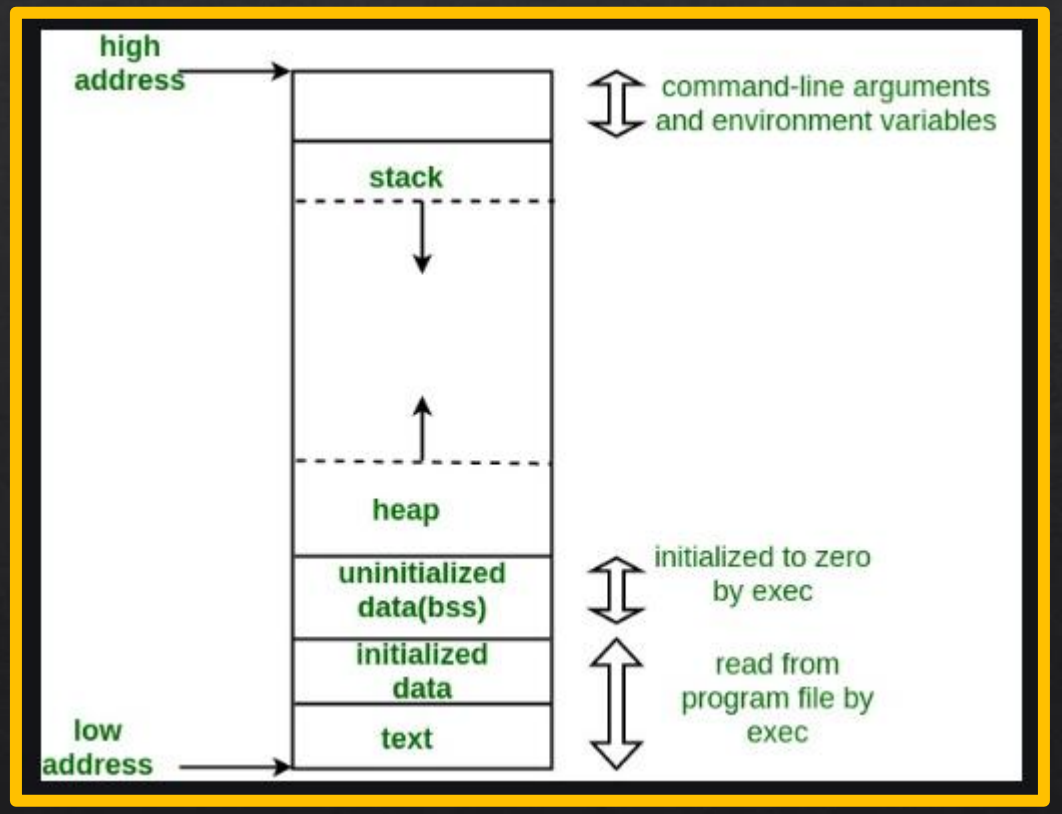
In 32-bit x86, the *base pointer* (formerly **%ebp**, now **%rbp**) was used to keep track of the base of the current stack frame, and a called function would save the base pointer of its caller prior to updating the base pointer to its own stack frame. With the advent of the 64-bit architecture, this has been mostly eliminated, save for a few special cases when the compiler cannot determine ahead of time how much stack space needs to be allocated for a particular function (see Dynamic stack allocation).

x64 Assembly -- Instructions

- `mov`
- `jmp`
- `call`
- `ret`
- `nop`
- …

Memory Layout

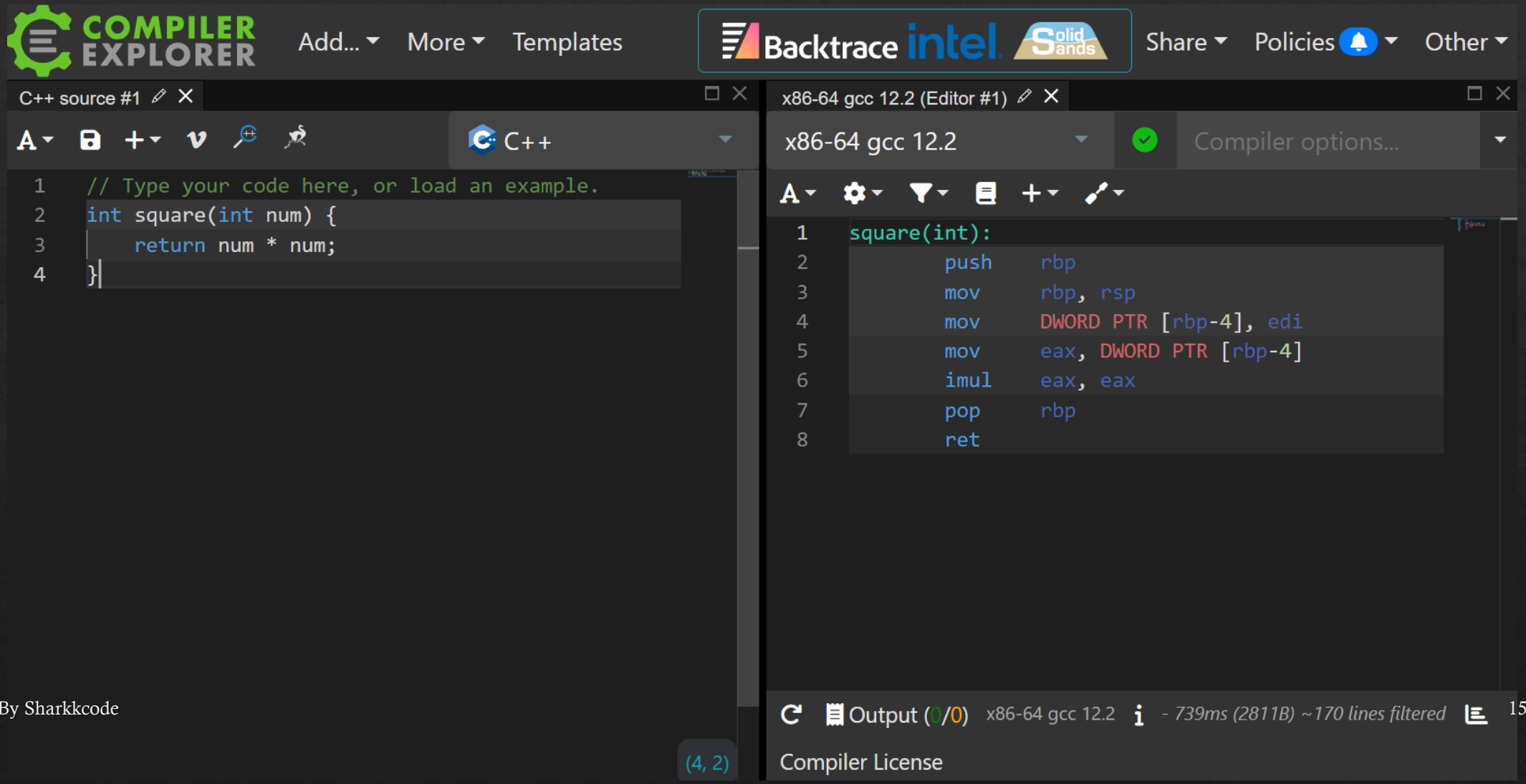
- https://www.geeksforgeeks.org/memory-layout-of-c-program/



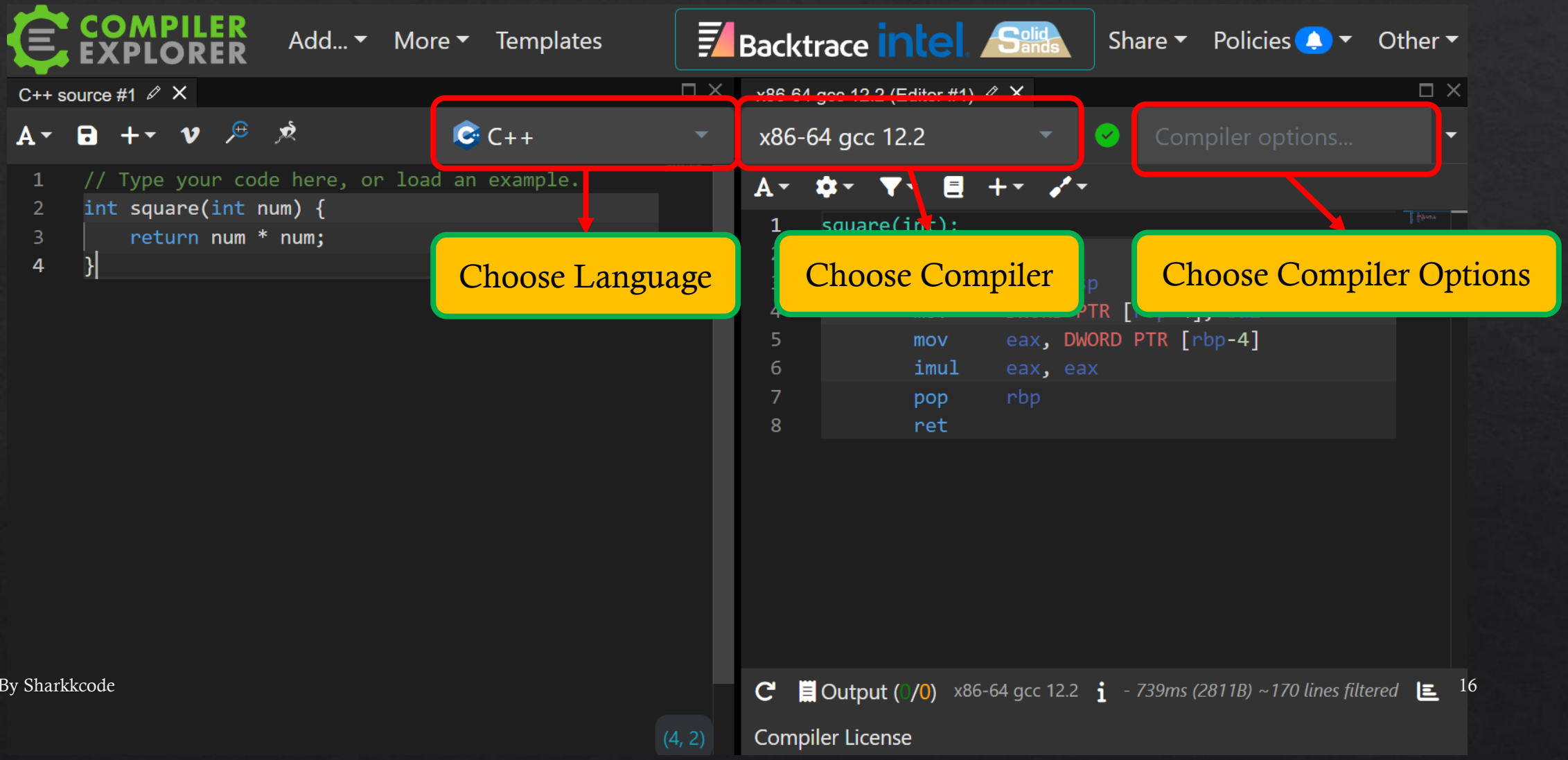
Demo Learning Website

https://godbolt.org/

Website Overview



Website Overview



Hello World DEMO

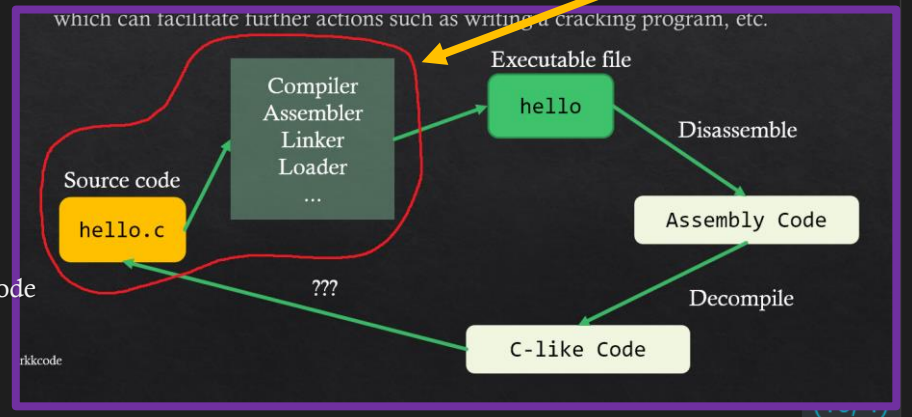
```c
// SOURCE CODE
#include <stdio.h>

int main(int argc, char **argv) {

    printf("Hello World!\n");

    return 0;
}
```

Hello World DEMO

COMPILER EXPLORER Add... More Templates Backtrace intel Solid Sands Share Policies Other

C++ source #1

```cpp
#include <stdio.h>

int main(int argc, char **argv) {

    printf("Hello World!\n");

    return 0;
}
```



x86-64 gcc 12.2 (Editor #1)

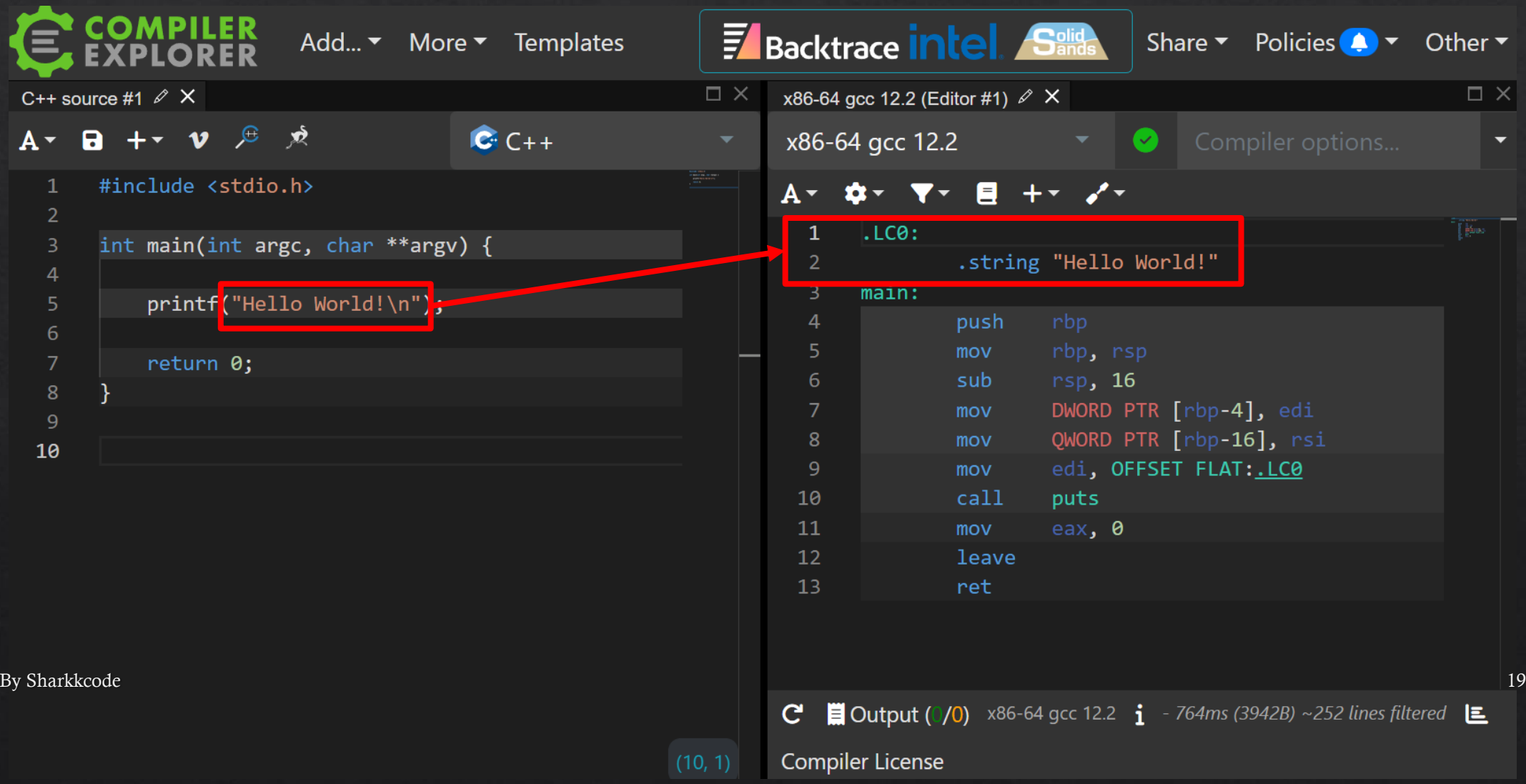
x86-64 gcc 12.2 Compiler options...

```asm
.LC0:
        .string "Hello World!"
main:
        push    rbp
        mov     rbp, rsp
        sub     rsp, 16
        mov     DWORD PTR [rbp-4], edi
        mov     QWORD PTR [rbp-16], rsi
        mov     edi, OFFSET FLAT:.LC0
        call    puts
        mov     eax, 0
        leave
        ret
```

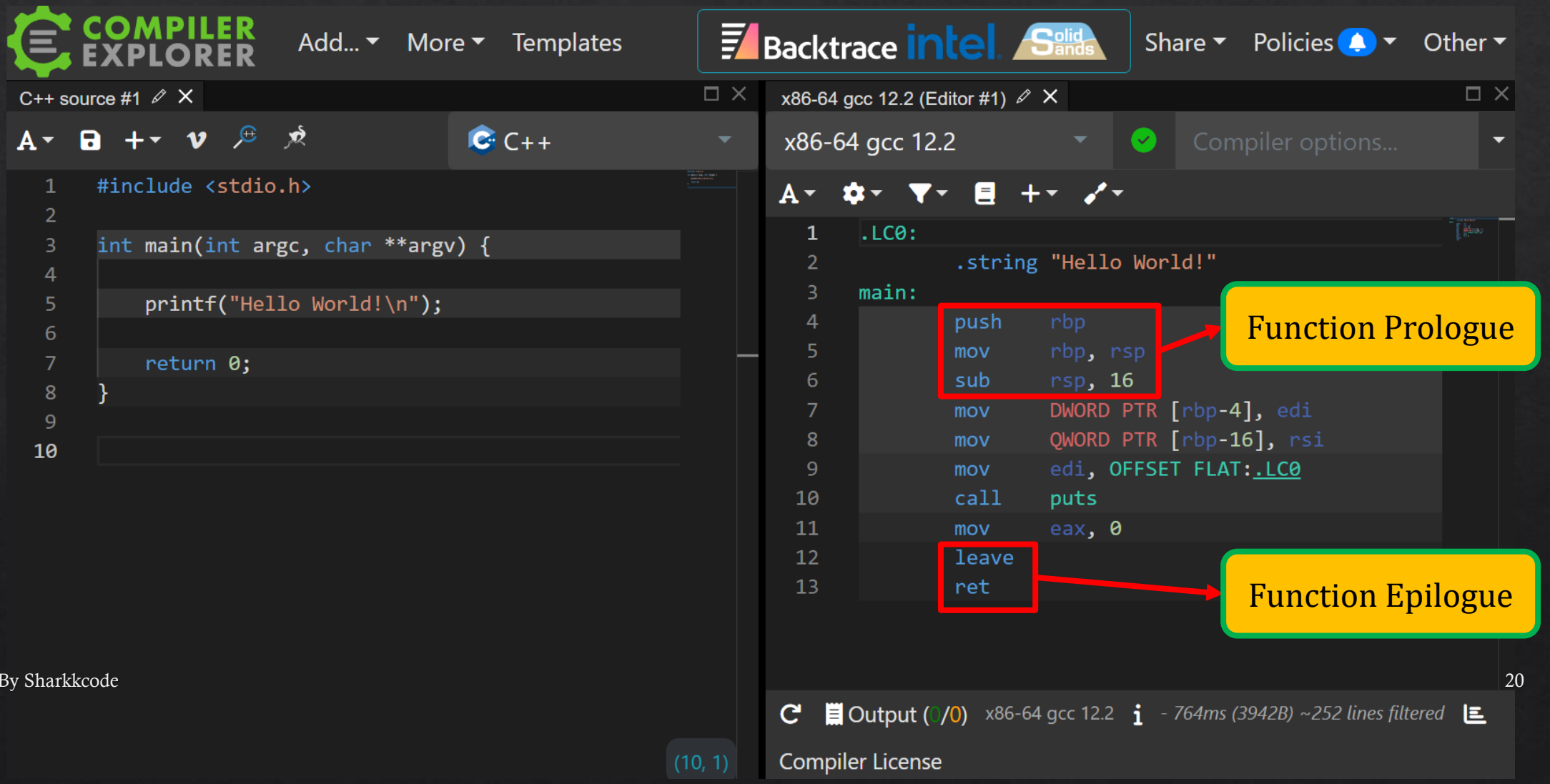
Output (0/0) x86-64 gcc 12.2 - 764ms (3942B) ~252 lines filtered

Compiler License

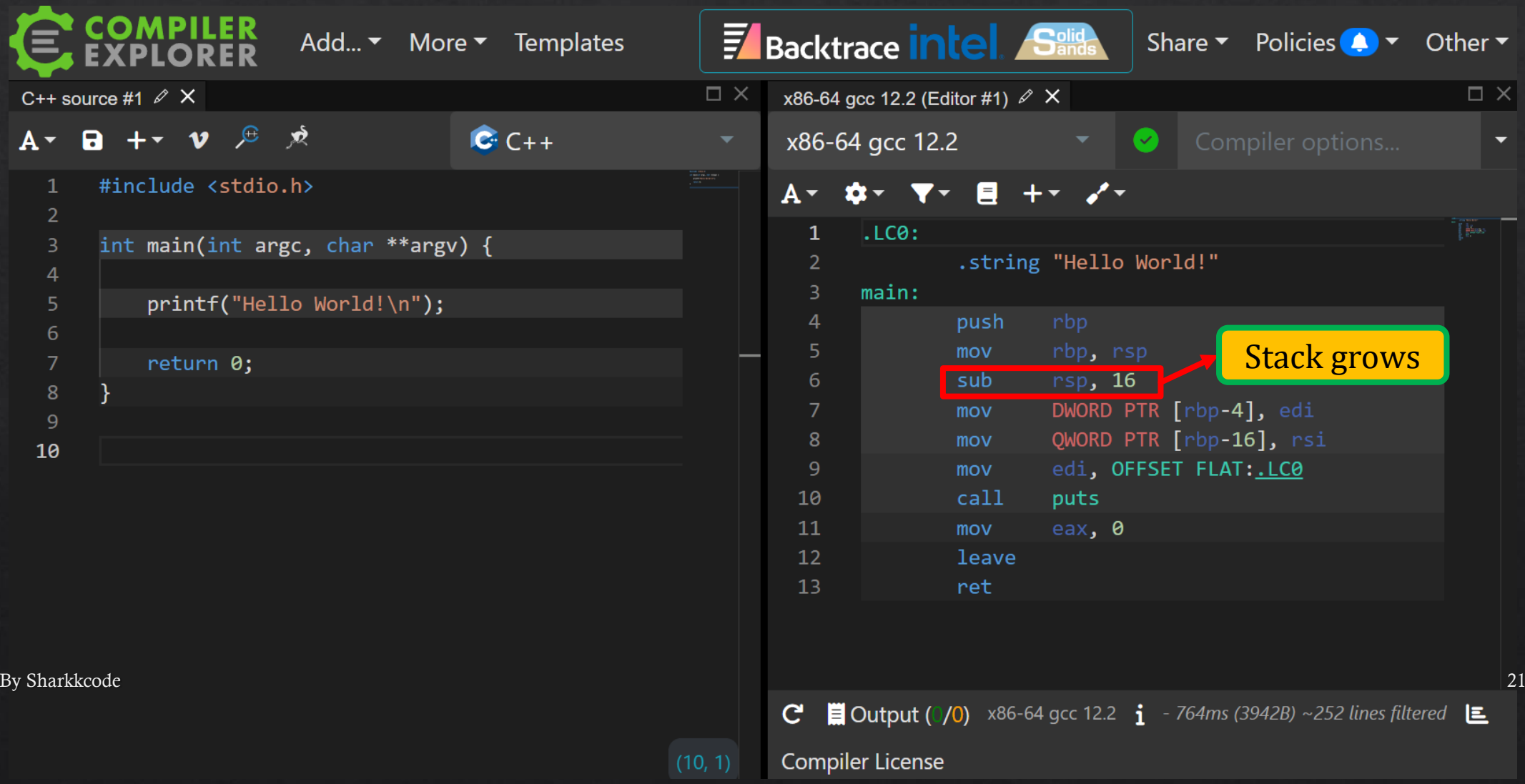
Hello World DEMO



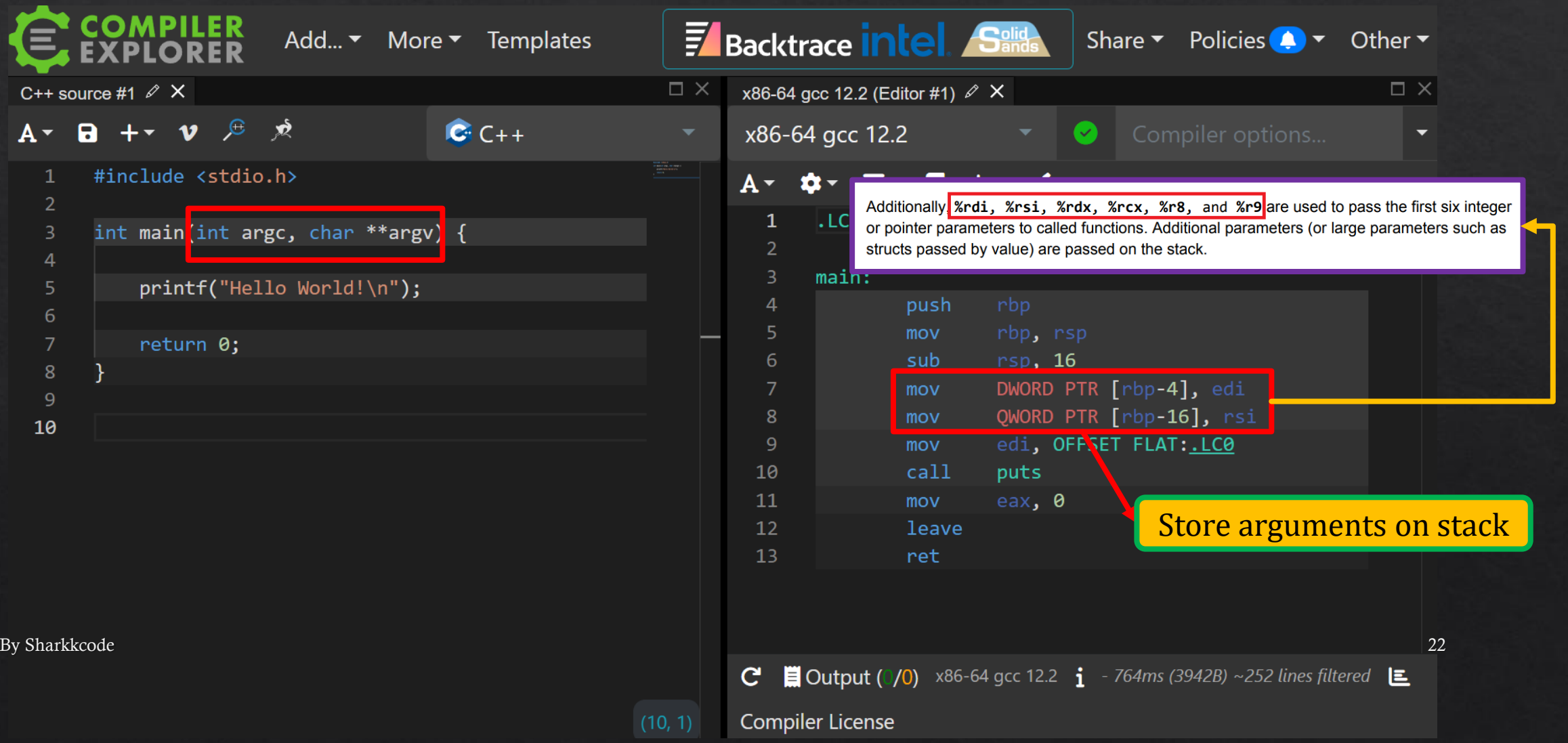
Hello World DEMO



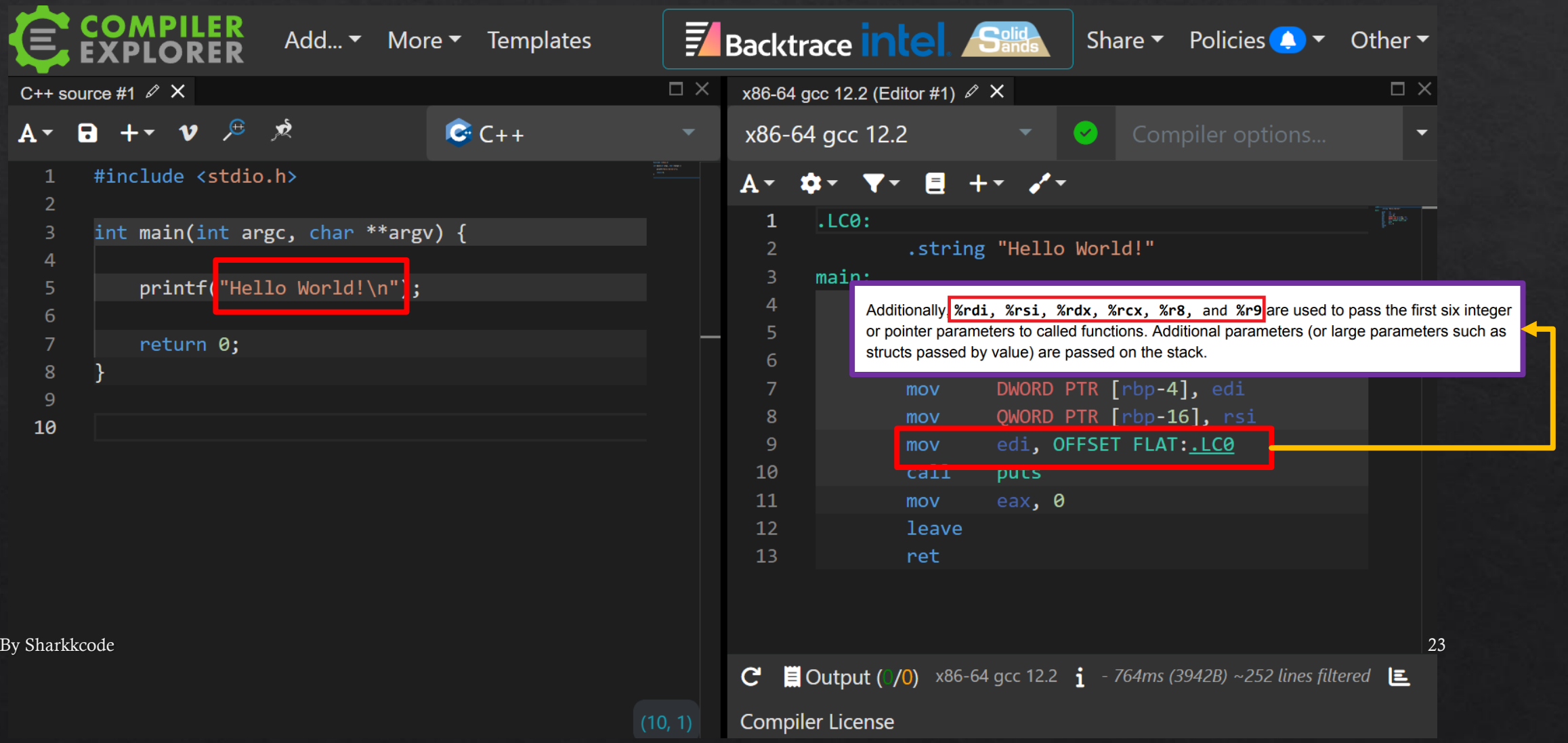
Hello World DEMO



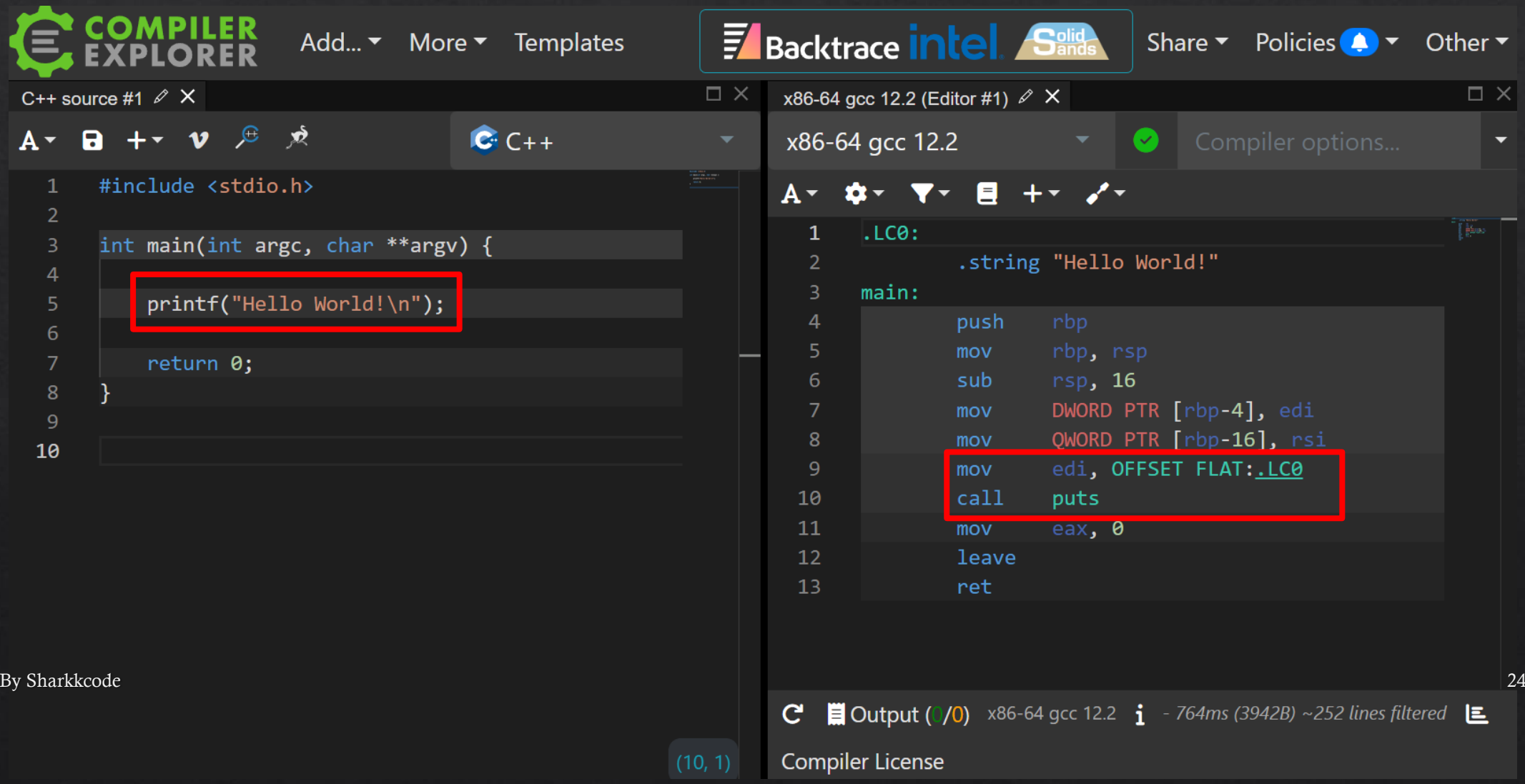
Hello World DEMO



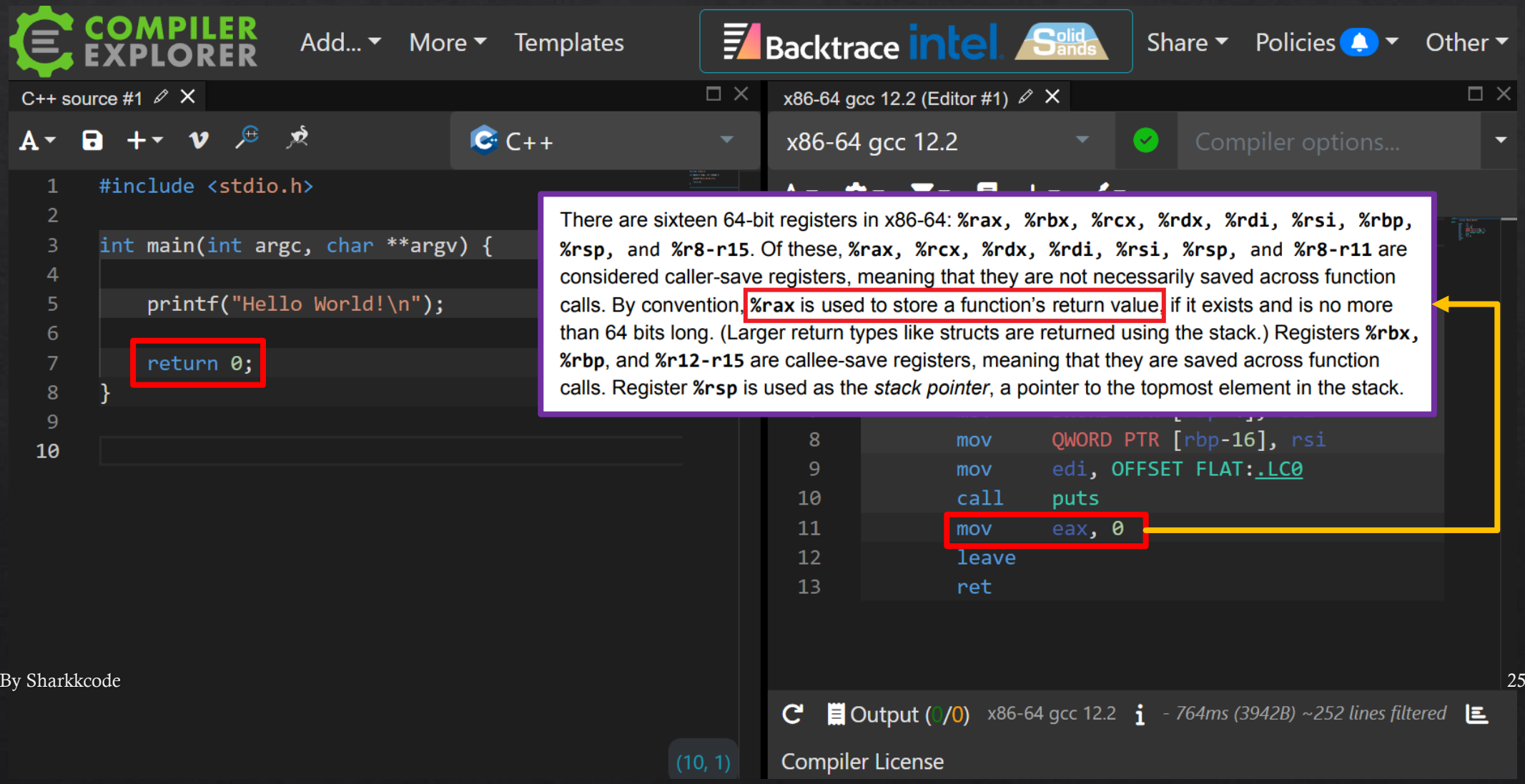
Hello World DEMO



Hello World DEMO



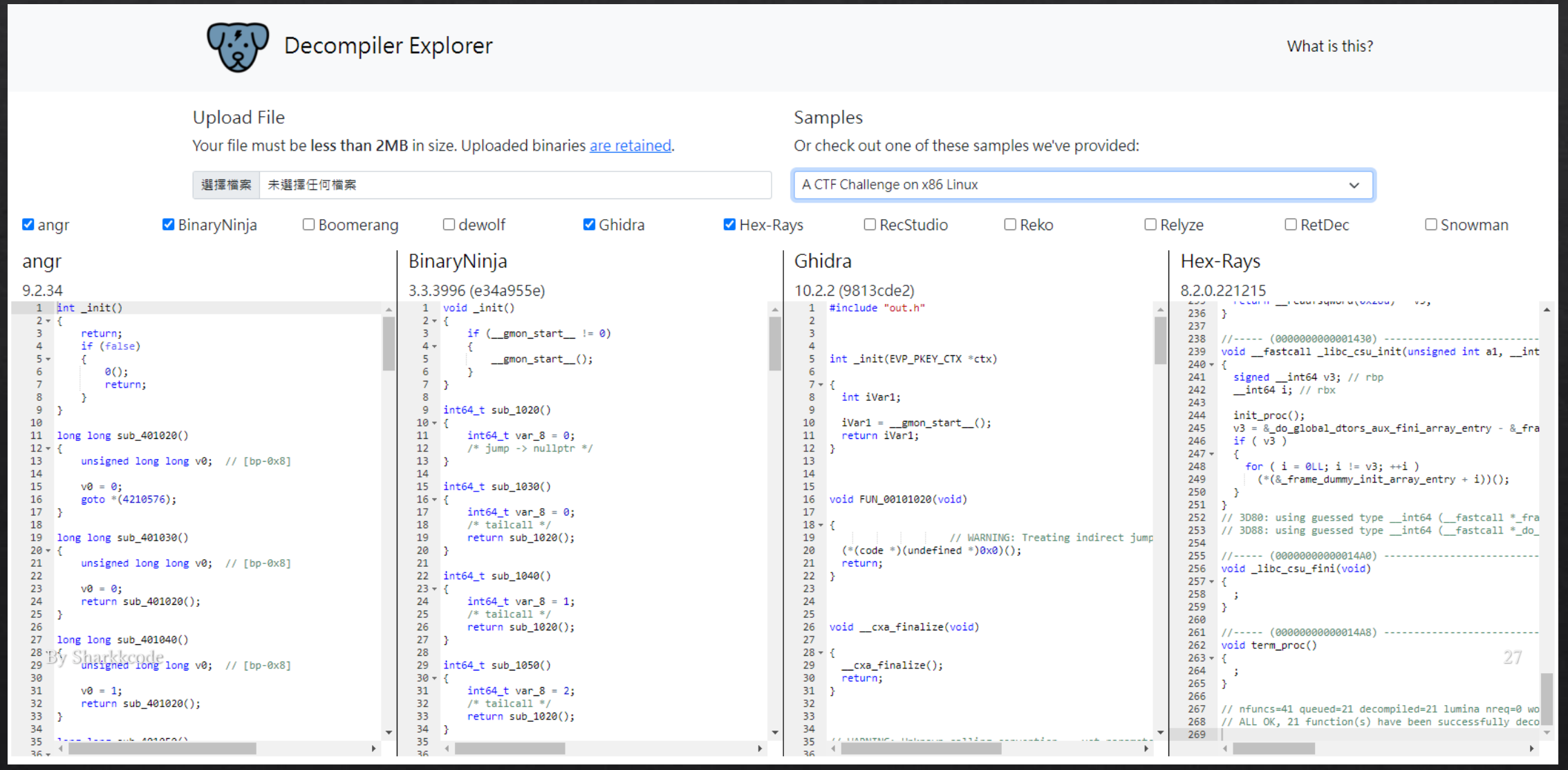
Hello World DEMO



omeD gninraeL etisbeW

https://dogbolt.org/

Compare with several decompilers



Lab

- https://play.picoctf.org/practice
  - ARMssembly 1

Tools

GDB, GHIDRA, pwntools

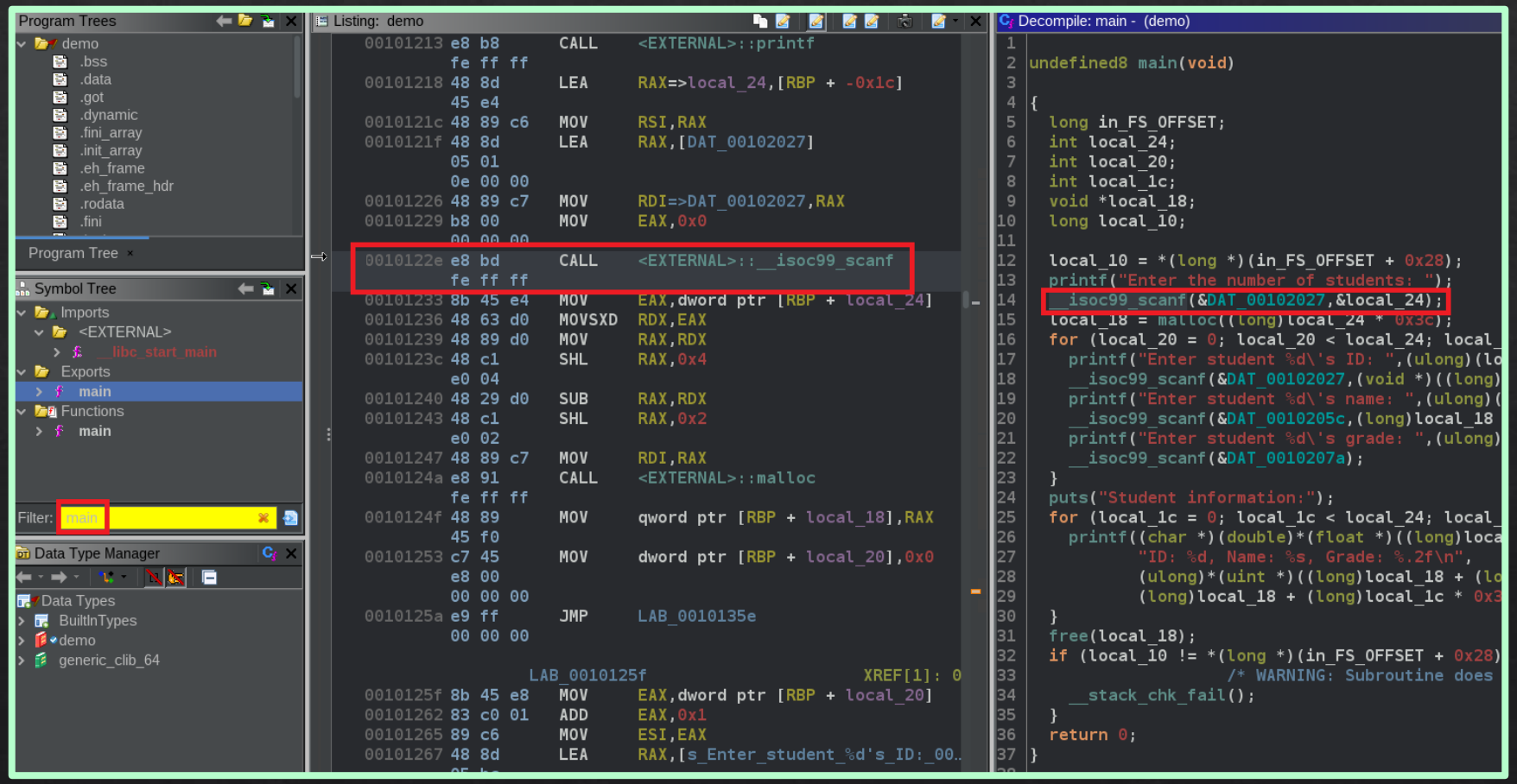
Tools

- demo.c

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int id;
    char name[50];
    float grade;
} Student;

int main() {

    setvbuf(stdout,0,2,0);
    setvbuf(stdin,0,2,0);
    setvbuf(stderr,0,2,0);

    int n;
    printf("Enter the number of students: ");
    scanf("%d", &n);
    Student *students = (Student*)malloc(n * sizeof(Student));
    for (int i = 0; i < n; i++) {
        printf("Enter student %d's ID: ", i+1);
        scanf("%d", &(students[i].id));
        printf("Enter student %d's name: ", i+1);
        scanf("%s", students[i].name);
        printf("Enter student %d's grade: ", i+1);
        scanf("%f", &(students[i].grade));
    }
    printf("Student information:\n");
    for (int i = 0; i < n; i++) {
        printf("ID: %d, Name: %s, Grade: %.2f\n", students[i].id, students[i].name, students[i].grade);
    }
    free(students);
    return 0;
}
```

Tools -- GHIDRA

- 22e



Tools -- GDB

- 22e

```
0x000055555555520e <+37>:    mov    eax,0x0
0x0000555555555213 <+42>:    call   0x5555555550d0 <printf@plt>
0x0000555555555218 <+47>:    lea    rax,[rbp-0x1c]
0x000055555555521c <+51>:    mov    rsi,rax
0x000055555555521f <+54>:    lea    rax,[rip±0xe01]        # 0x555555556027
0x0000555555555226 <+61>:    mov    rdi,rax
0x0000555555555229 <+64>:    mov    eax,0x0
0x000055555555522e <+69>:    call   0x5555555550f0 <__isoc99_scanf@plt>
0x0000555555555233 <+74>:    mov    eax,DWORD PTR [rbp-0x1c]
0x0000555555555236 <+77>:    movsxd rdx,eax
0x0000555555555239 <+80>:    mov    rax,rdx
0x000055555555523c <+83>:    shl    rax,0x4
0x0000555555555240 <+87>:    sub    rax,rdx
0x0000555555555243 <+90>:    shl    rax,0x2
0x0000555555555247 <+94>:    mov    rdi,rax
0x000055555555524a <+97>:    call   0x5555555550e0 <malloc@plt>
0x000055555555524f <+102>:   mov    QWORD PTR [rbp-0x10],rax
0x0000555555555253 <+106>:   mov    DWORD PTR [rbp-0x18],0x0
```

Tools -- pwntools

- demo_py.py

```python
from pwn import *

io = process("./demo")

io.sendafter(b": ", b"1\n")

io.sendafter(b": ", b"2\n")
io.sendafter(b": ", b"3\n")
io.sendafter(b": ", b"4\n")

io.interactive()

```

Tools -- pwntools

- `python3 demo_py.py`

```
~/rev_pwn python3 demo_py.py
[+] Starting local process './demo': pid 1741811
[*] Switching to interactive mode
[*] Process './demo' stopped with exit code 0 (pid 1741811)
Student information:
ID: 2, Name: 3, Grade: 4.00
[*] Got EOF while reading in interactive
$
```

Lab

./reverse_demo_s

Author Sharkkcode

Your goal is to find out the right key!

Lab

./tea

From SHELL CTF 2022

Your goal is to find out the FLAG!

THANKS

Q&A