Reverse Engineering

2024/8/6 國立清華大學資訊安全研究所暑訓

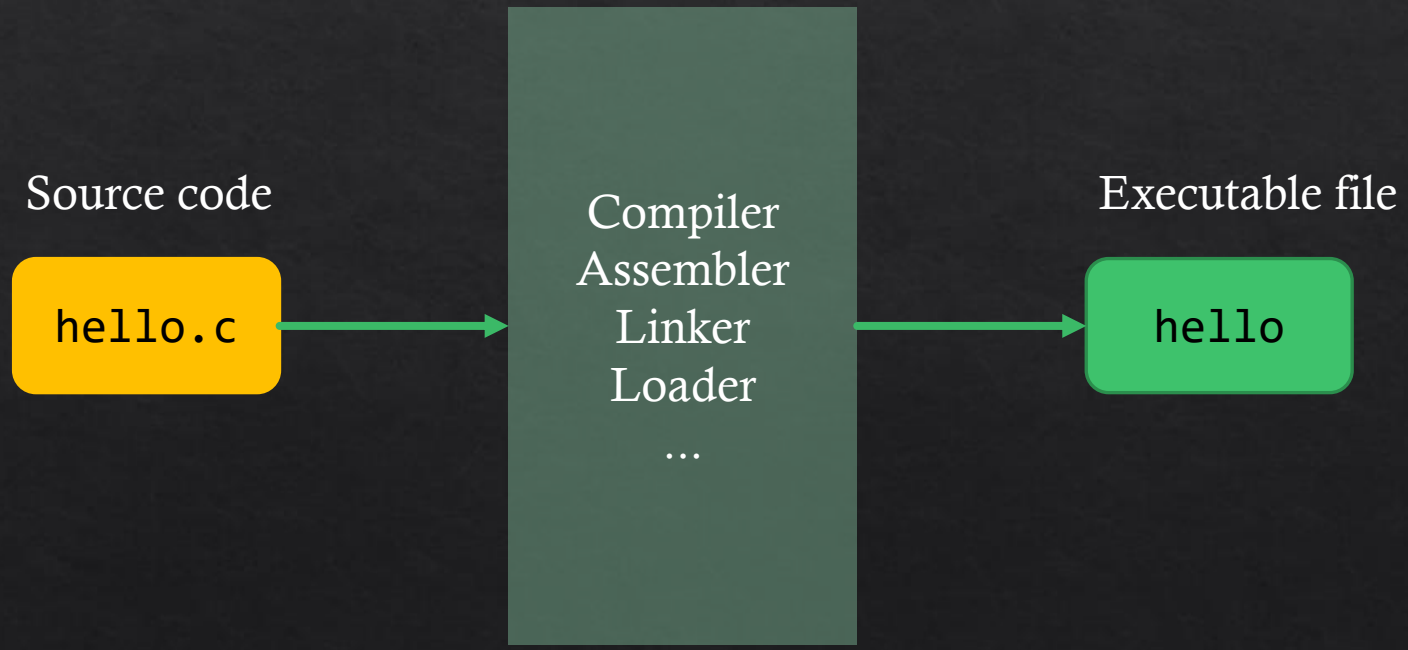
潘甫翰 Sharkkcode

Outline

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

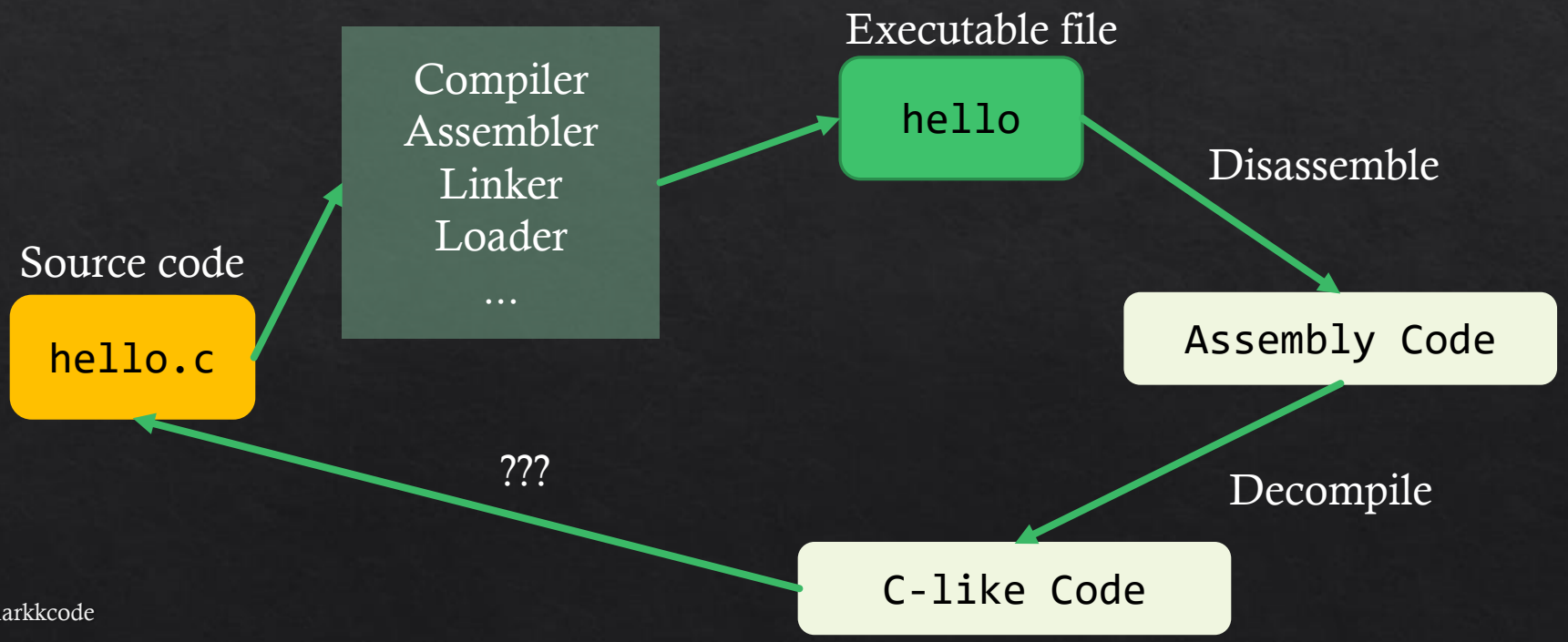
Reverse Engineering Overview

- 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…

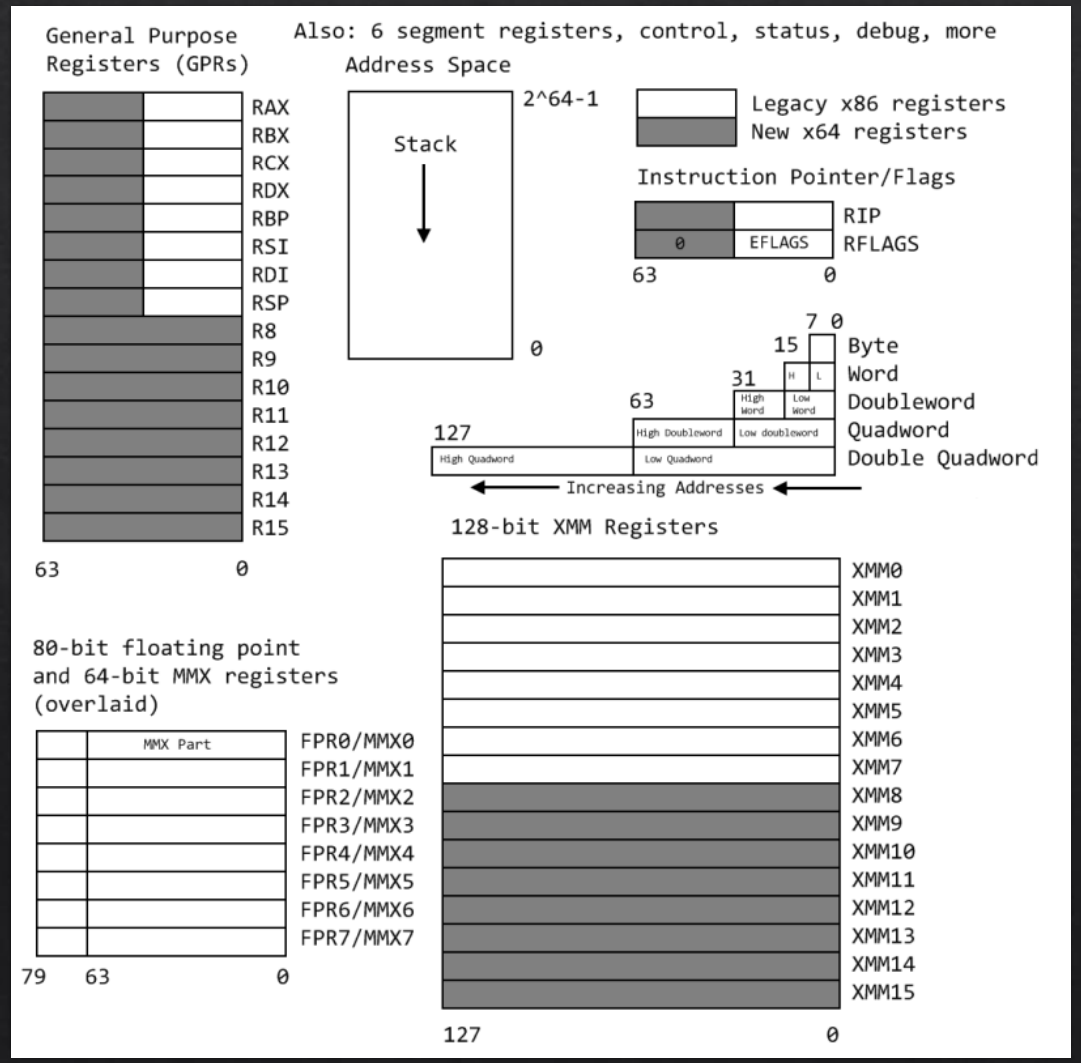
x64 Assembly

- 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.

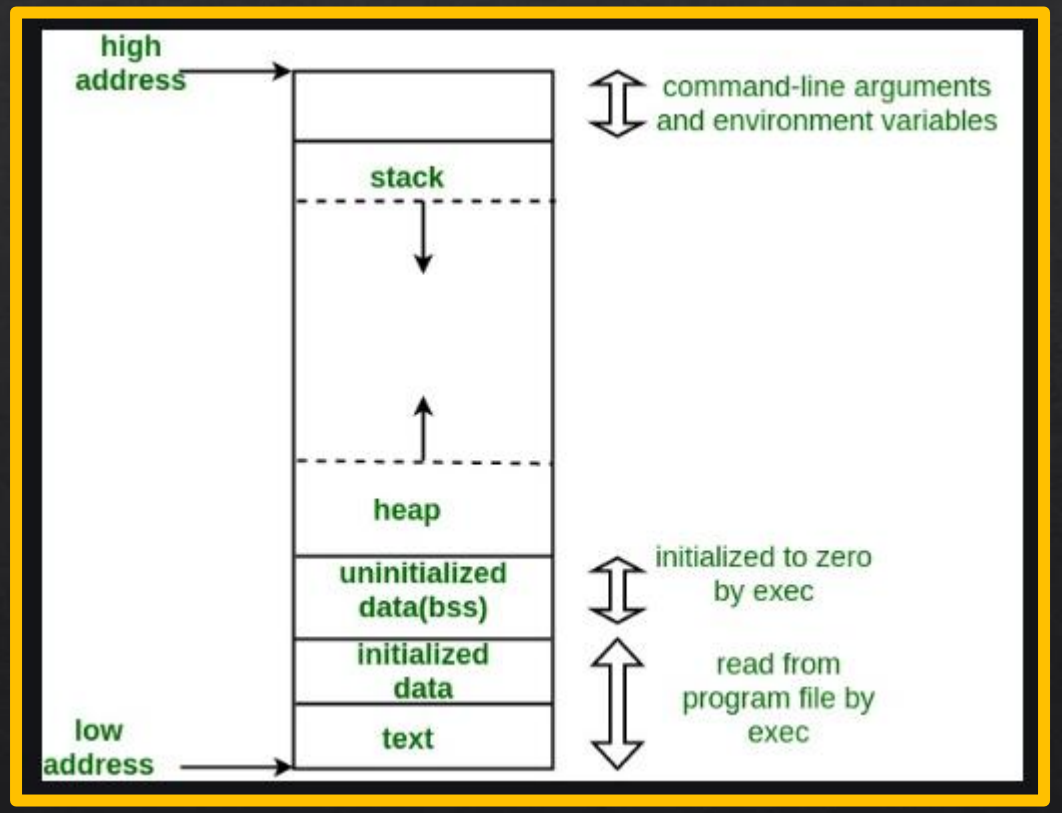
In 32-bit x86, the *base pointer* (formerly **%ebp**, now **%rbp**) was used to keep track of the base of the current stack frame, and a called function would save the base pointer of its caller prior to updating the base pointer to its own stack frame. With the advent of the 64-bit architecture, this 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

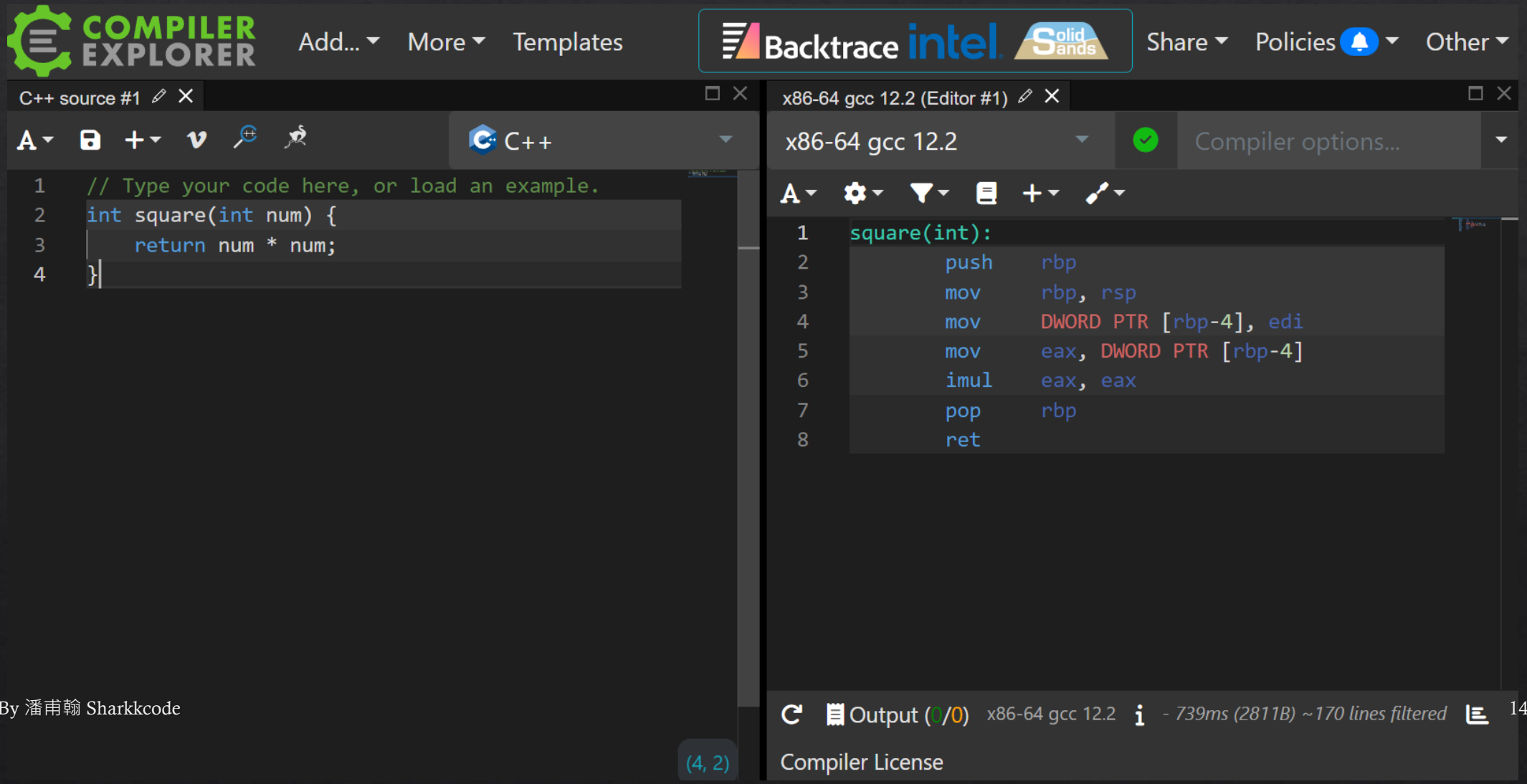
- https://www.geeksforgeeks.org/memory-layout-of-c-program/



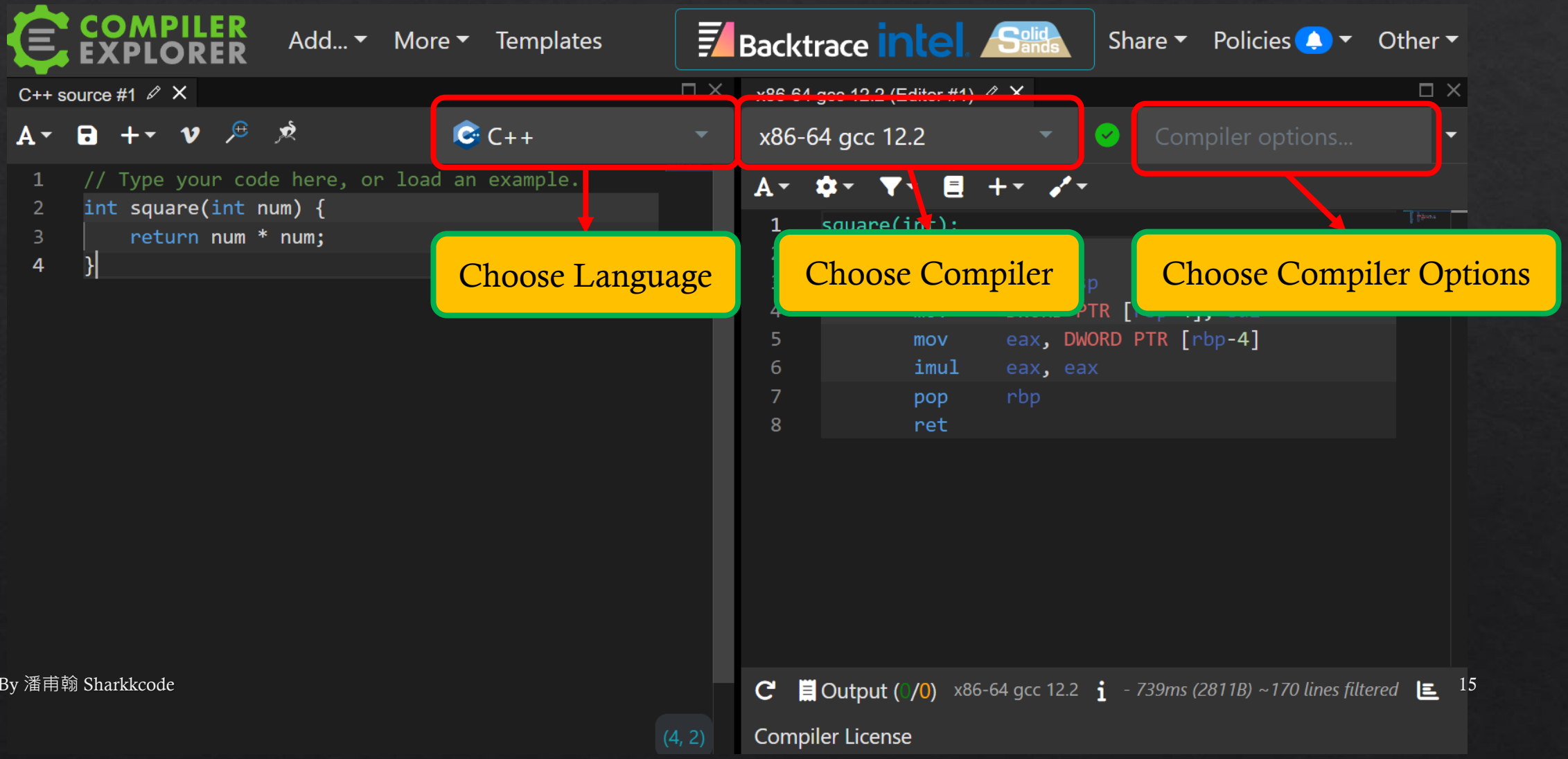
Demo Learning Website

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

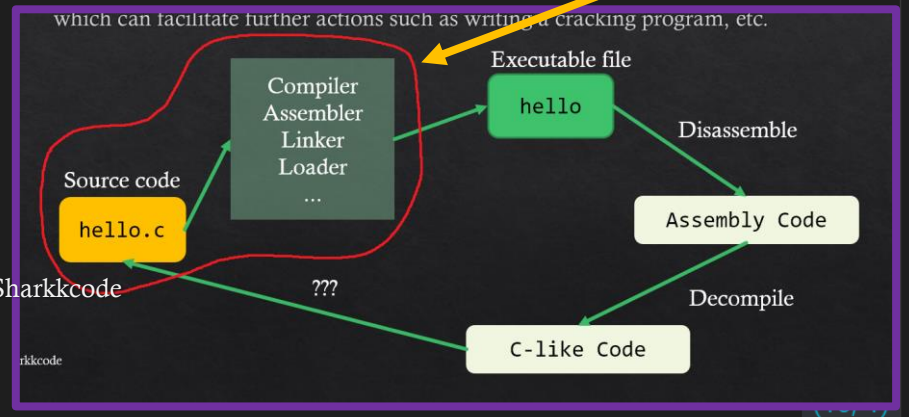
C++ source #1

```
1  #include <stdio.h>
2
3  int main(int argc, char **argv) {
4
5      printf("Hello World!\n");
6
7      return 0;
8  }
9
10
```

x86-64 gcc 12.2 (Editor #1)

x86-64 gcc 12.2 Compiler options...

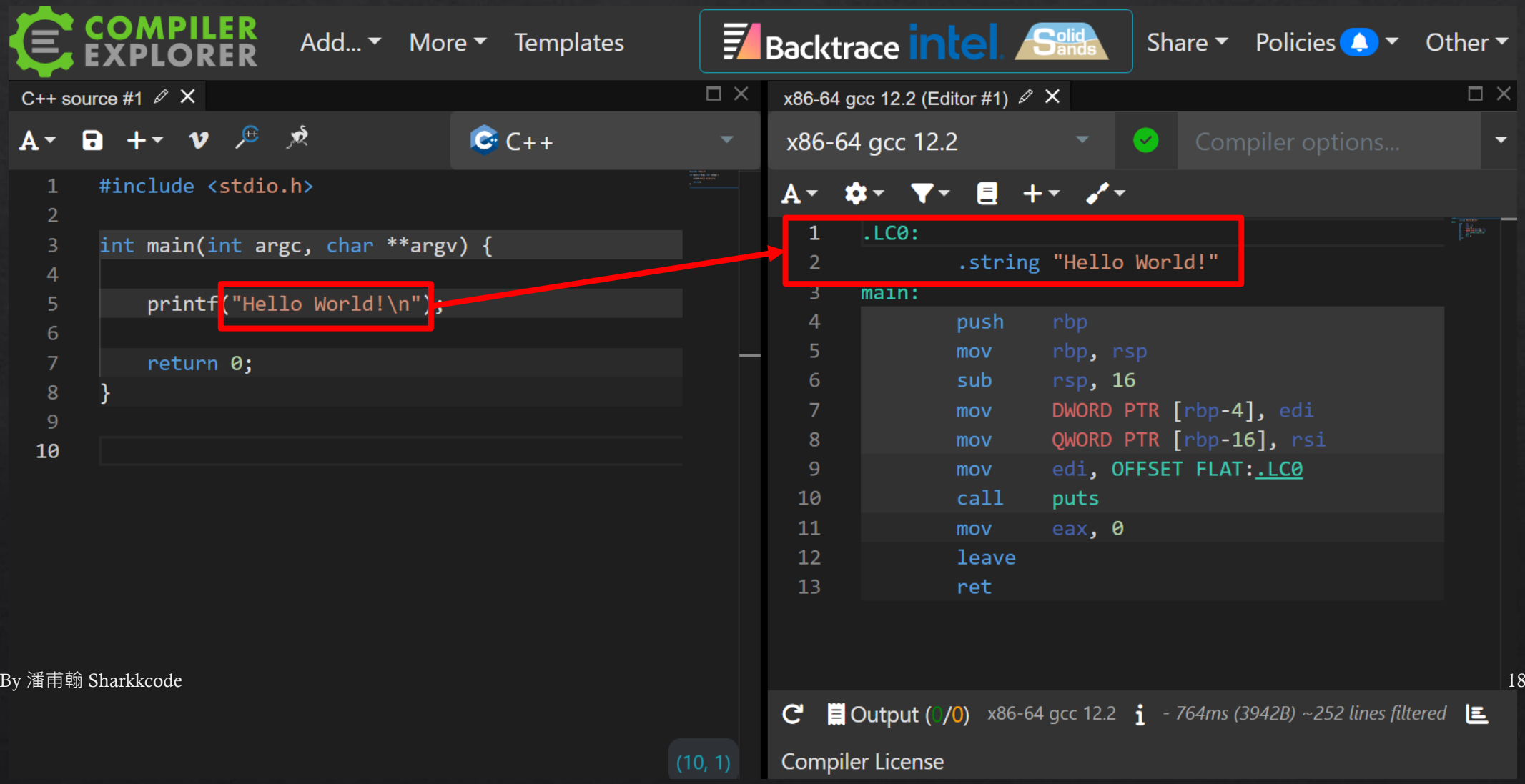
```
1   .LC0:
2           .string "Hello World!"
3   main:
4           push    rbp
5           mov     rbp, rsp
6           sub     rsp, 16
7           mov     DWORD PTR [rbp-4], edi
8           mov     QWORD PTR [rbp-16], rsi
9           mov     edi, OFFSET FLAT:.LC0
10          call    puts
11          mov     eax, 0
12          leave
13          ret
```



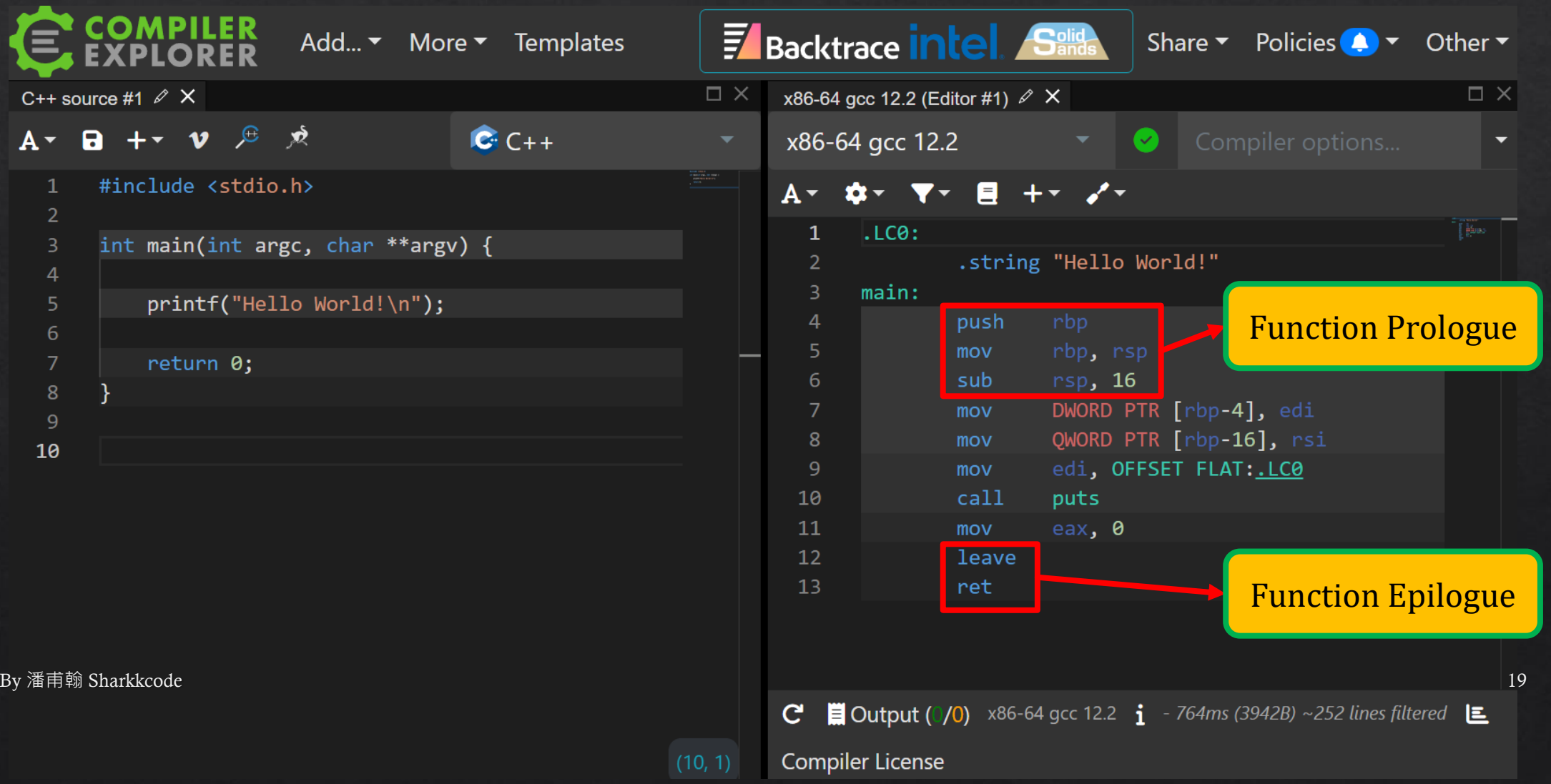
Output (0/0) x86-64 gcc 12.2 - 764ms (3942B) ~252 lines filtered

Compiler License

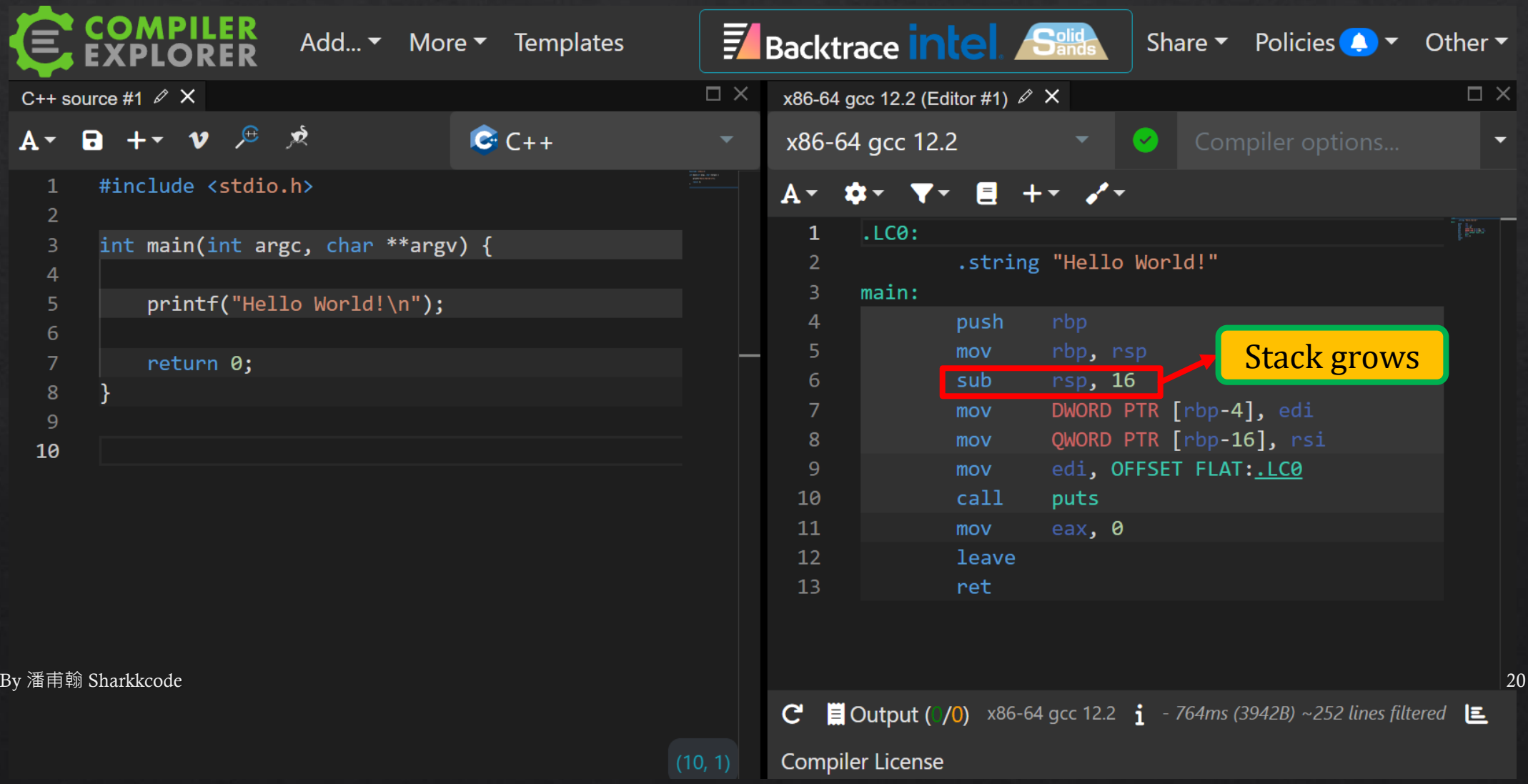
Hello World DEMO



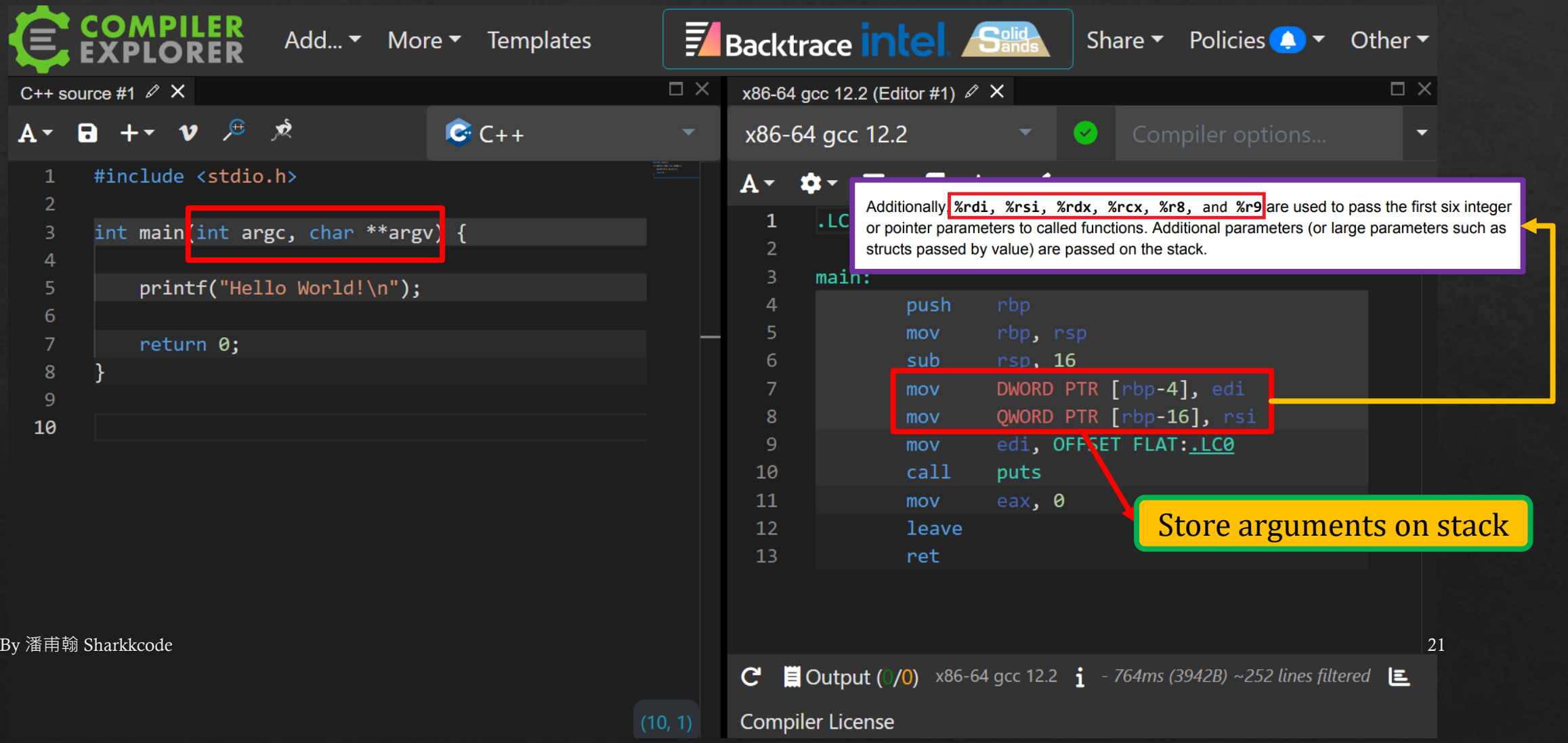
Hello World DEMO



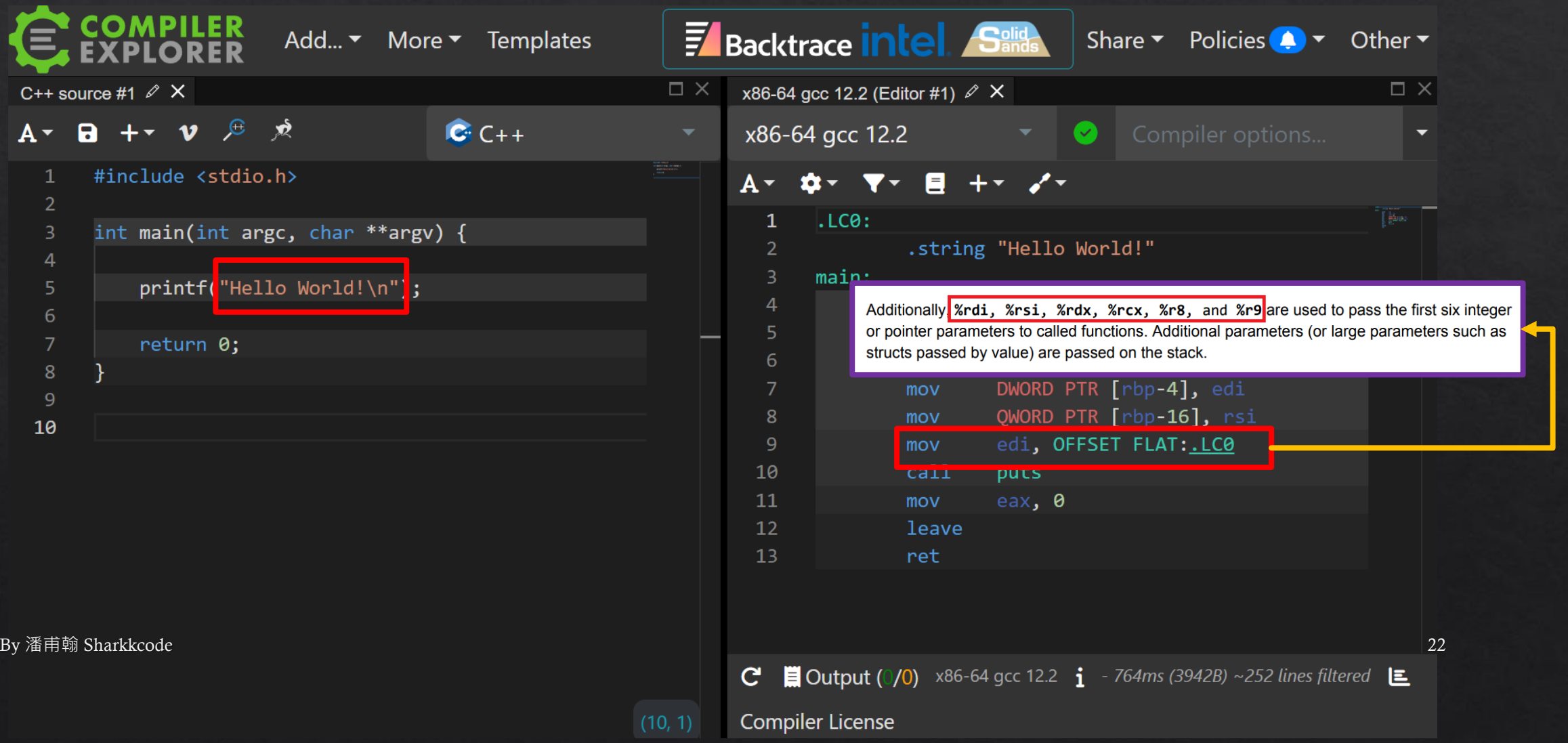
Hello World DEMO



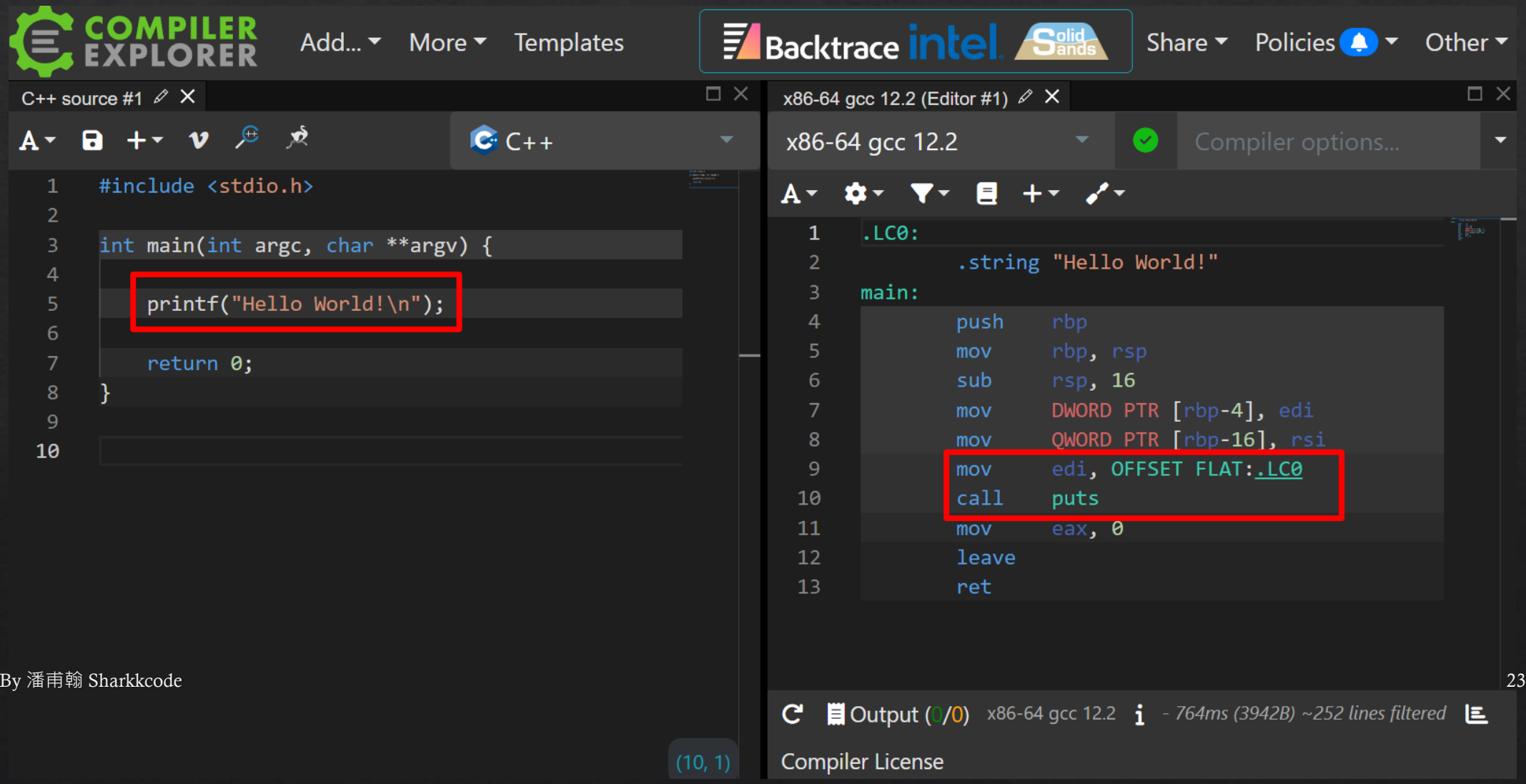
Hello World DEMO



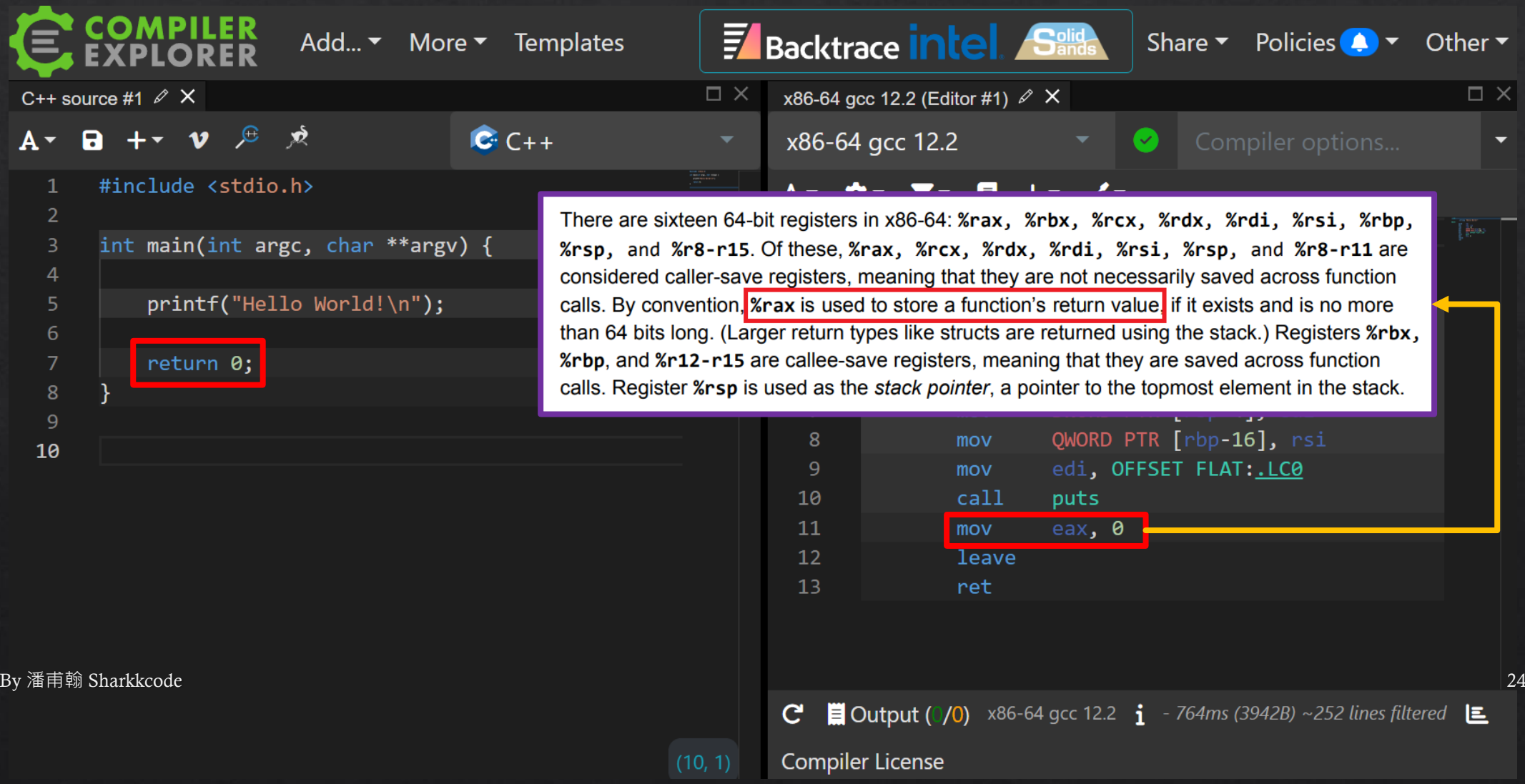
Hello World DEMO



Hello World DEMO



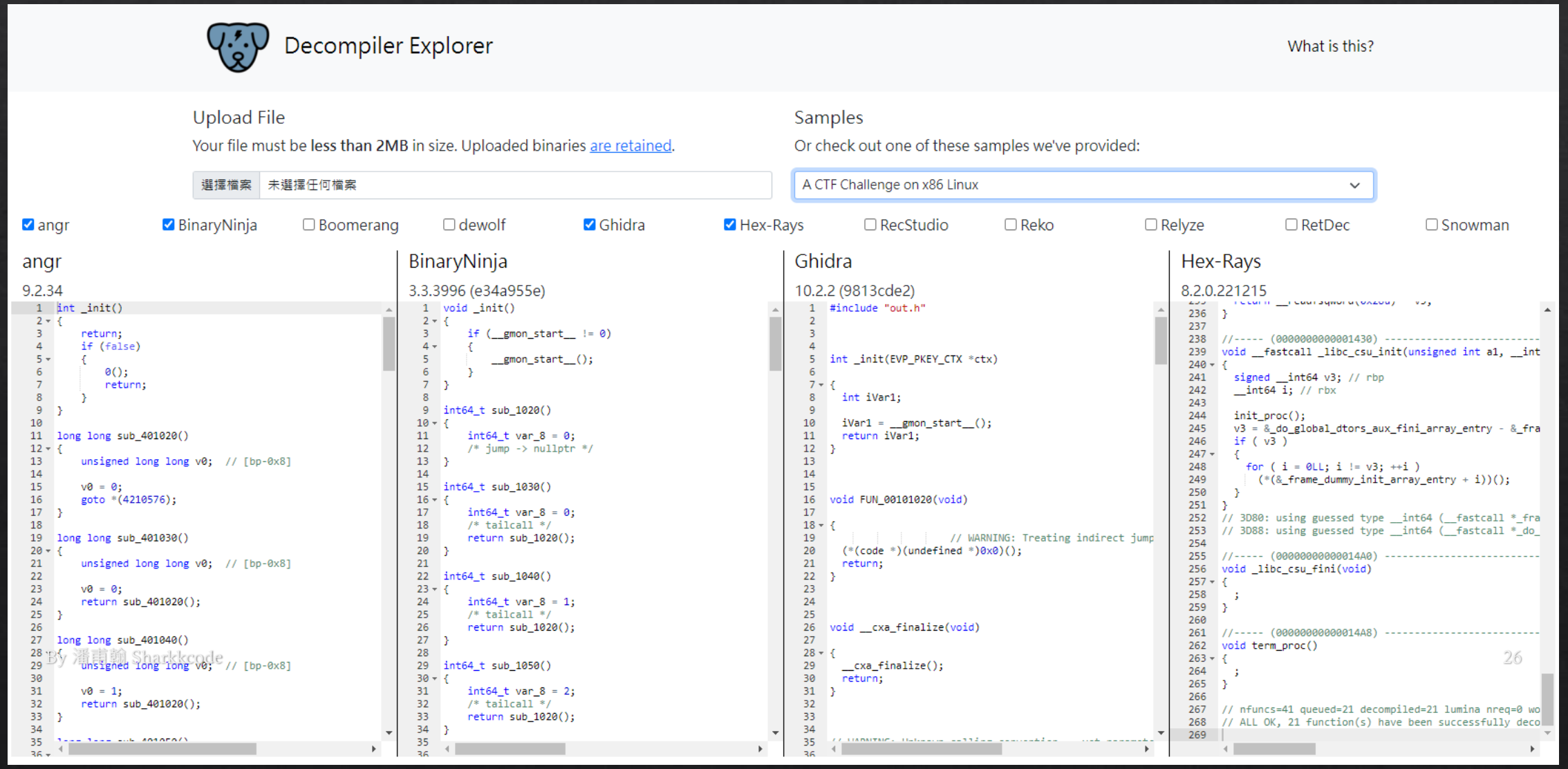
Hello World DEMO



omeD gninraeL etisbeW

https://dogbolt.org/

Compare with several decompilers




```
int _init()
{
    return;
    if (false)
    {
        0();
        return;
    }
}

long long sub_401020()
{
    unsigned long long v0;  // [bp-0x8]

    v0 = 0;
    goto *(4210576);
}

long long sub_401030()
{
    unsigned long long v0;  // [bp-0x8]

    v0 = 0;
    return sub_401020();
}

long long sub_401040()
{
    unsigned long long v0;  // [bp-0x8]

    v0 = 1;
    return sub_401020();
}
```

```
void _init()
{
    if (__gmon_start__ != 0)
    {
        __gmon_start__();
    }
}

int64_t sub_1020()
{
    int64_t var_8 = 0;
    /* jump -> nullptr */
}

int64_t sub_1030()
{
    int64_t var_8 = 0;
    /* tailcall */
    return sub_1020();
}

int64_t sub_1040()
{
    int64_t var_8 = 1;
    /* tailcall */
    return sub_1020();
}

int64_t sub_1050()
{
    int64_t var_8 = 2;
    /* tailcall */
    return sub_1020();
}
```

```
#include "out.h"


int _init(EVP_PKEY_CTX *ctx)

{
  int iVar1;

  iVar1 = __gmon_start__();
  return iVar1;
}



void FUN_00101020(void)

{
                    // WARNING: Treating indirect jump
  (*(code *)(undefined *)0x0)();
  return;
}



void __cxa_finalize(void)

{
  __cxa_finalize();
  return;
}
```

```
}

//----- (0000000000001430) ----------------------------------------
void __fastcall _libc_csu_init(unsigned int a1, __int
{
  signed __int64 v3; // rbp
  __int64 i; // rbx

  init_proc();
  v3 = &_do_global_dtors_aux_fini_array_entry - &_fra
  if ( v3 )
  {
    for ( i = 0LL; i != v3; ++i )
      (*(&_frame_dummy_init_array_entry + i))();
  }
}
// 3D80: using guessed type __int64 (__fastcall *_fra
// 3D88: using guessed type __int64 (__fastcall *_do_

//----- (00000000000014A0) ----------------------------------------
void _libc_csu_fini(void)
{
  ;
}

//----- (00000000000014A8) ----------------------------------------
void term_proc()
{
  ;
}

// nfuncs=41 queued=21 decompiled=21 lumina nreq=0 wo
// ALL OK, 21 function(s) have been successfully deco
```

Lab

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

ARMssembly 1

Chall_1.S

- armv8

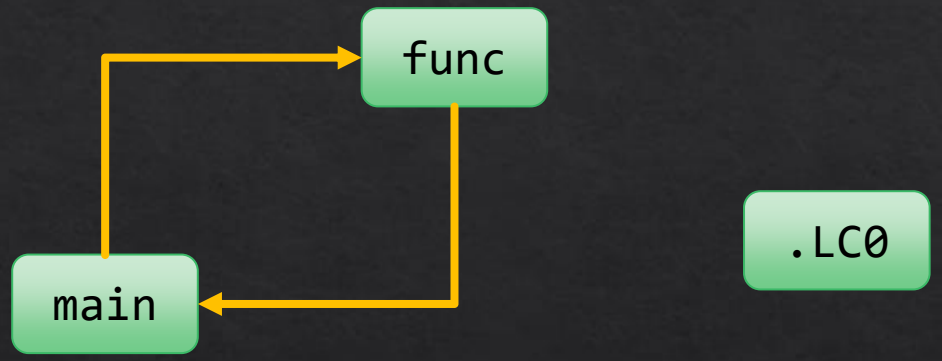
```
.arch armv8-a
.file   "chall_1.c"
.text
.align  2
.global func
.type   func, %function
```

Chall_1.S

- Program Flow

Chall_1.S

- Program Flow



.LC0

.LC1

.L4

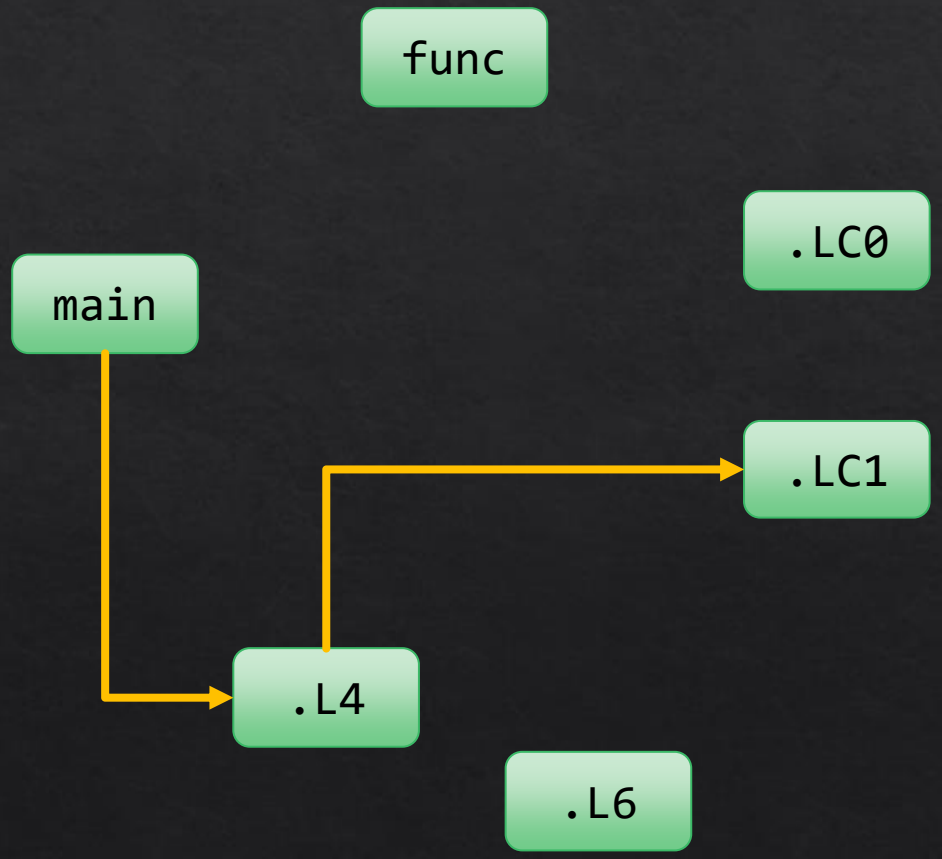
.L6

Chall_1.S

- Program Flow

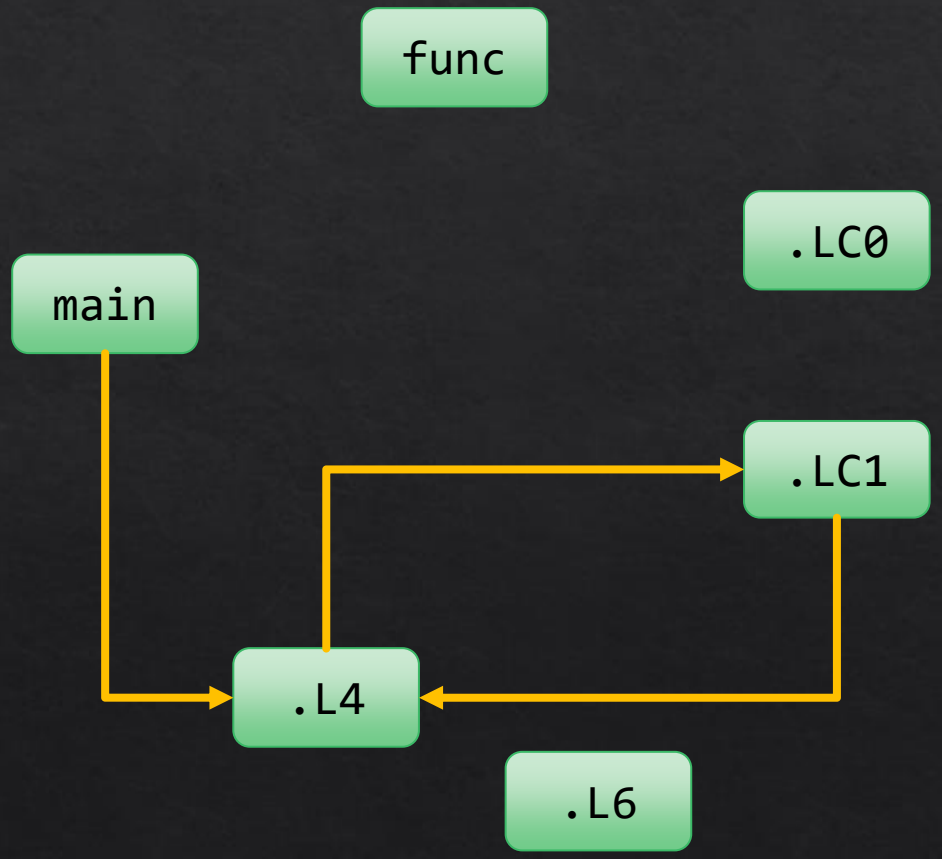
Chall_1.S

- Program Flow



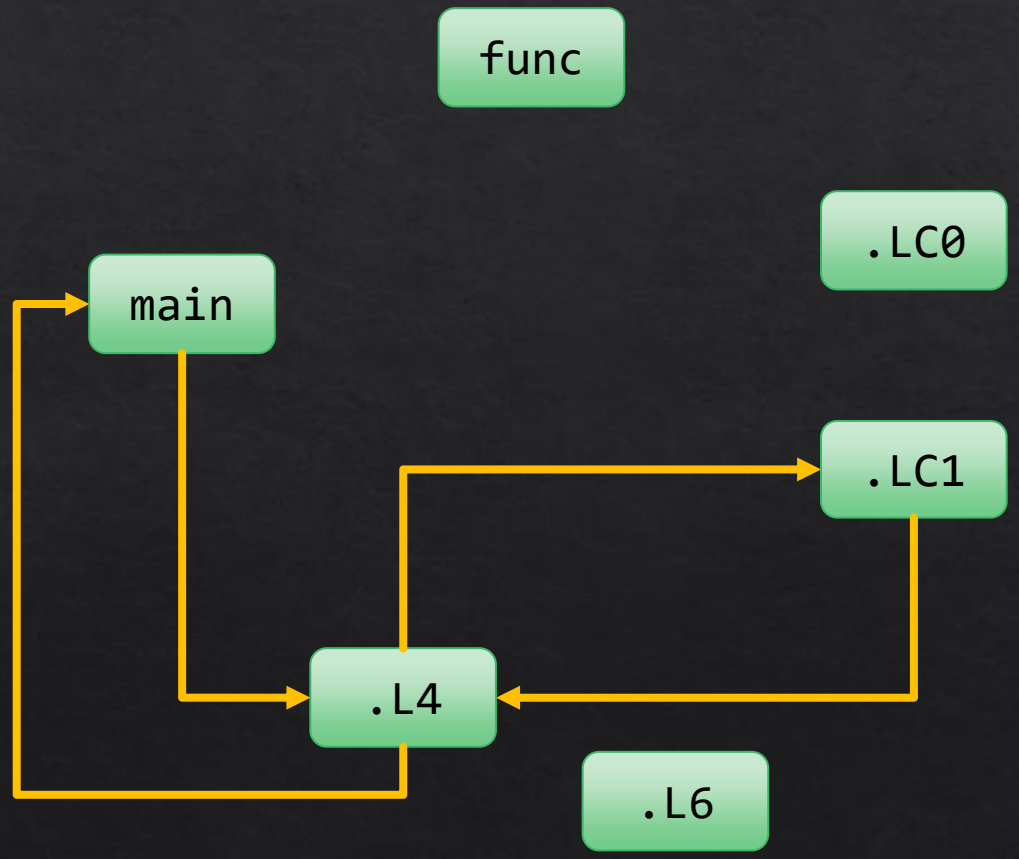
Chall_1.S

- Program Flow



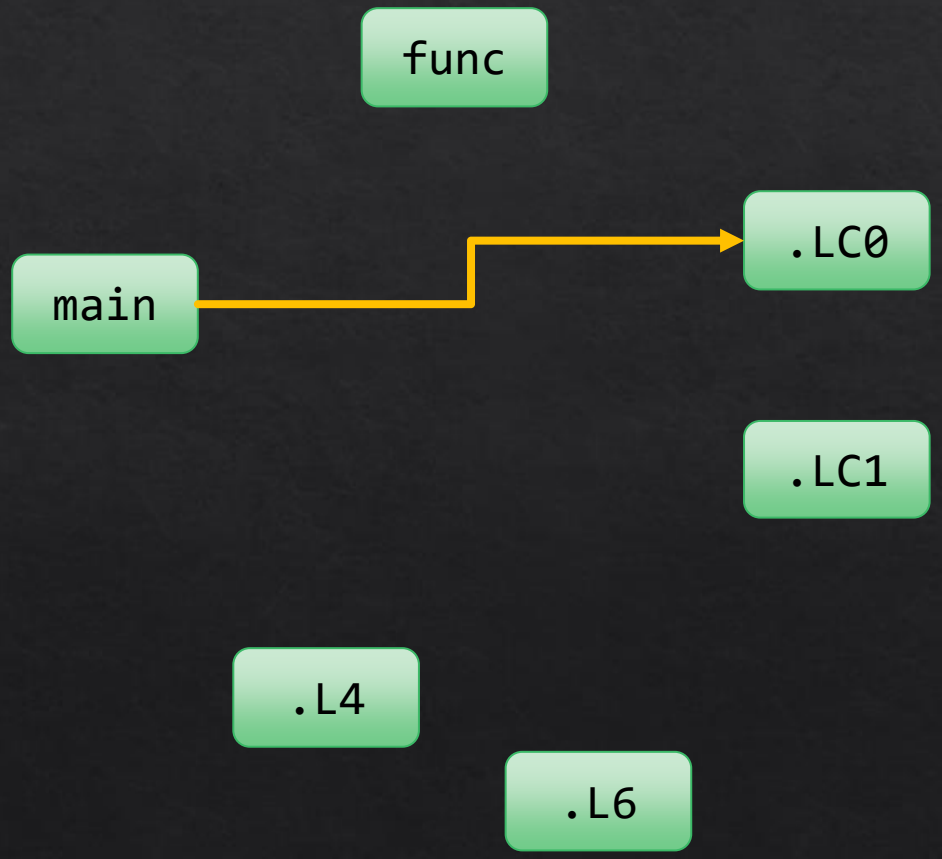
Chall_1.S

- Program Flow



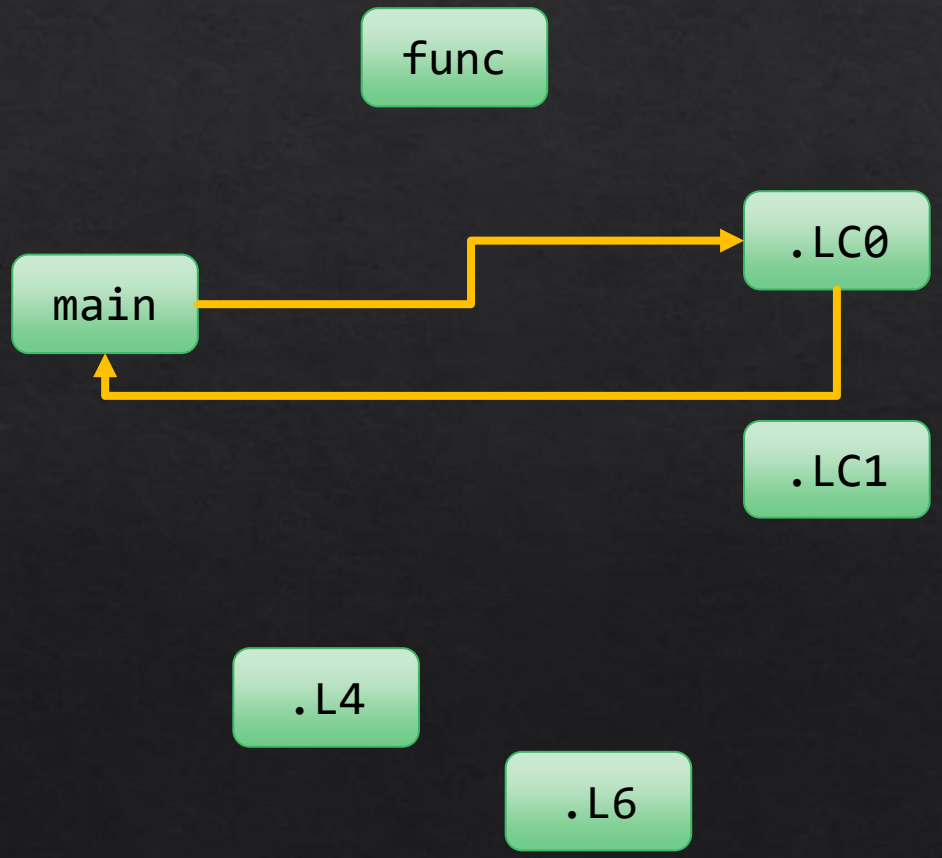
Chall_1.S

- Program Flow



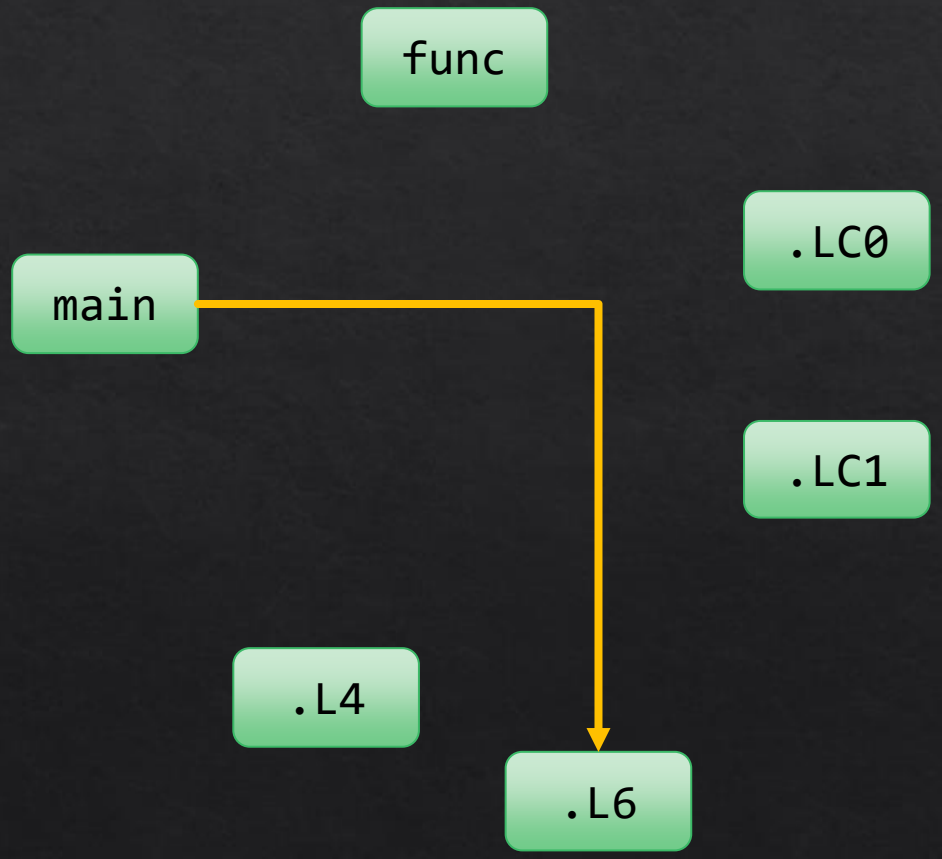
Chall_1.S

- Program Flow



Chall_1.S

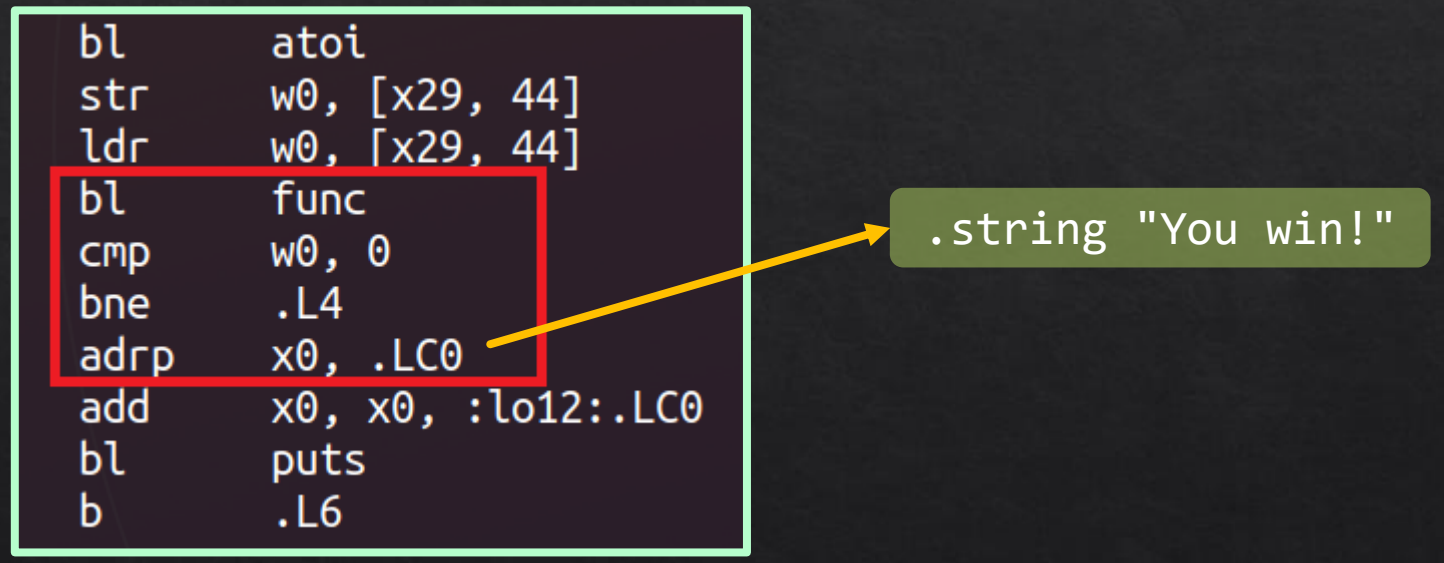
- Program Flow



Chall_1.S

- How to `win` ?
  - When `func` returns 0 (`w0 == 0`) .

```
bl      atoi
str     w0, [x29, 44]
ldr     w0, [x29, 44]
bl      func
cmp     w0, 0
bne     .L4
adrp    x0, .LC0
add     x0, x0, :lo12:.LC0
bl      puts
b       .L6
```



Chall_1.S

- func

Stack

???

Register : Value

SP	
PC	
w0	x
w1	y

```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

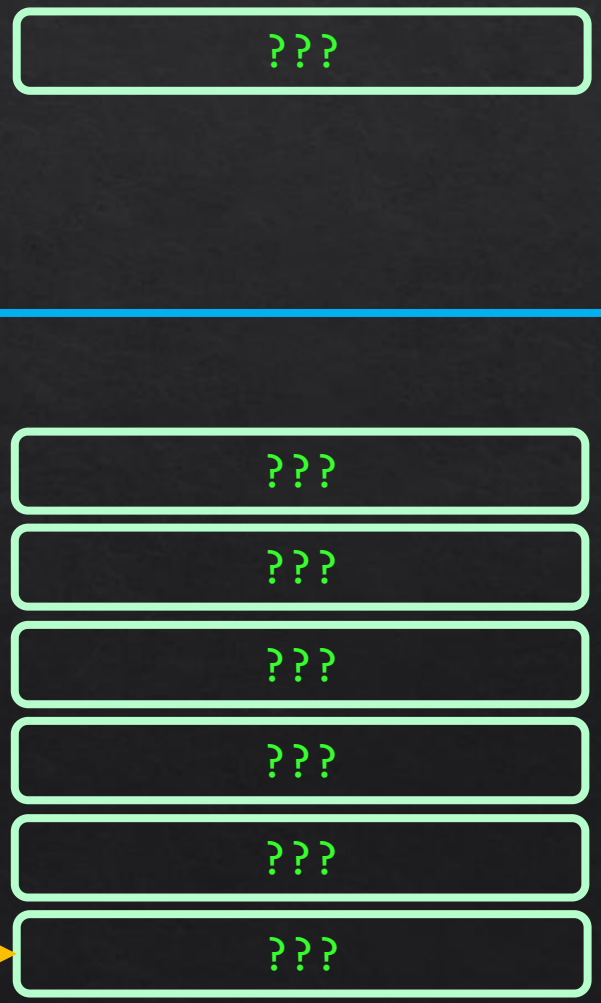
Chall_1.S

- func

Register : Value

SP	
PC	
w0	x
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

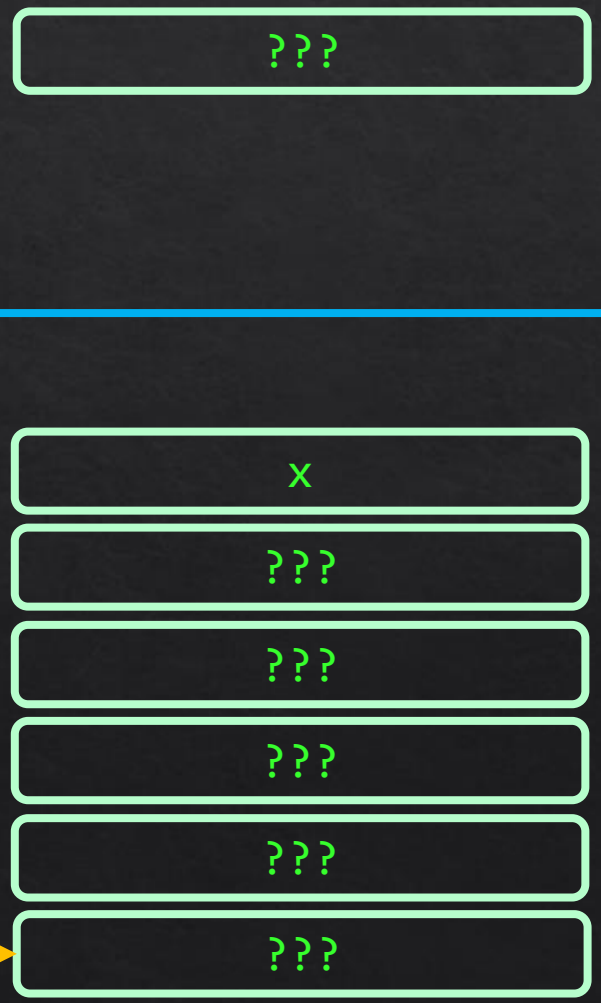
Chall_1.S

- func

Register : Value

SP	
PC	
w0	x
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

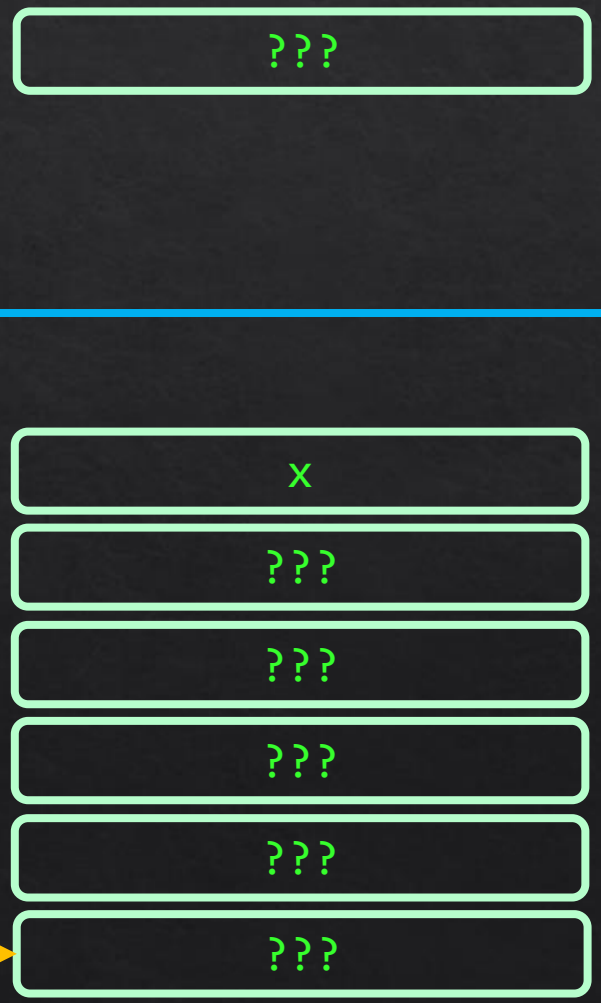
Chall_1.S

- func

Register : Value

SP	
PC	
w0	68
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

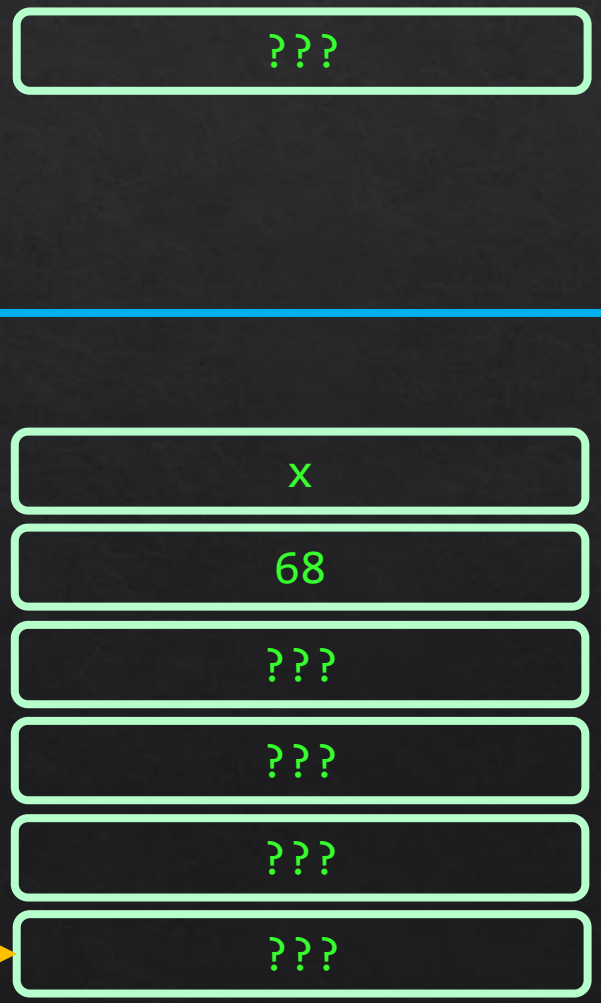
Chall_1.S

- func

Register : Value

SP	
PC	
w0	68
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

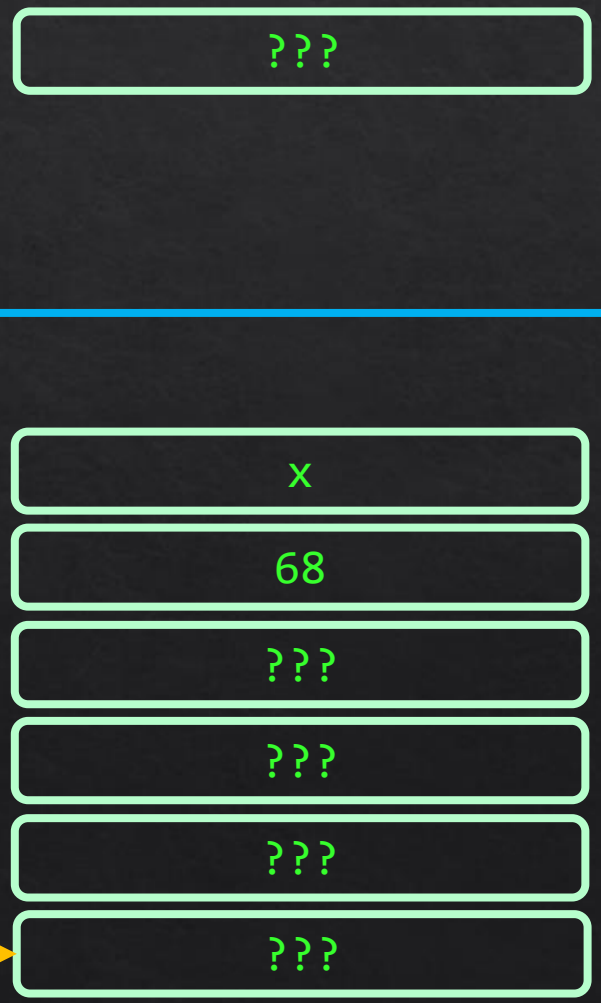
Chall_1.S

- func

Register : Value

SP	
PC	
w0	2
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

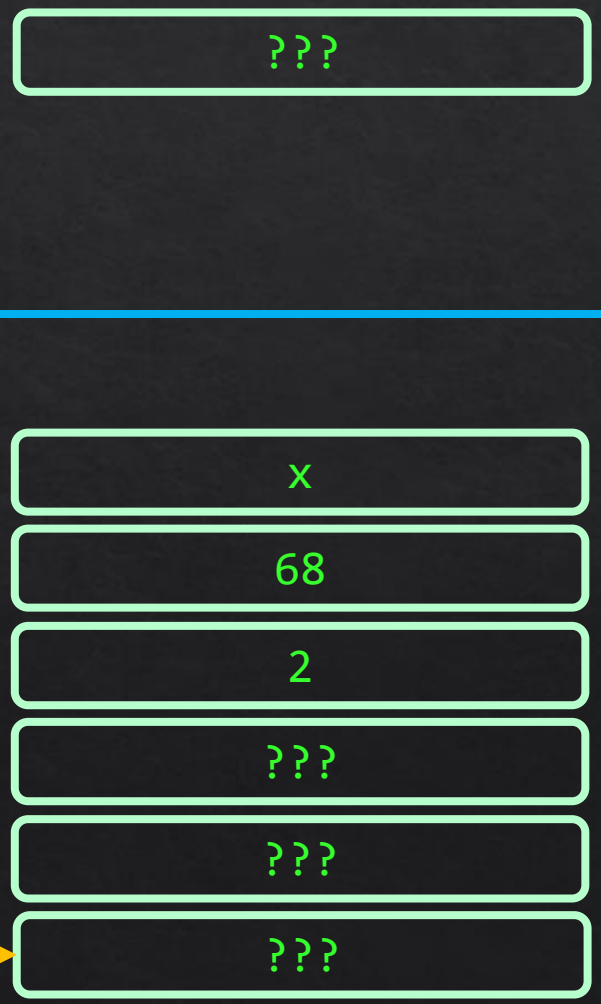
Chall_1.S

- func

Register : Value

SP	
PC	
w0	2
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

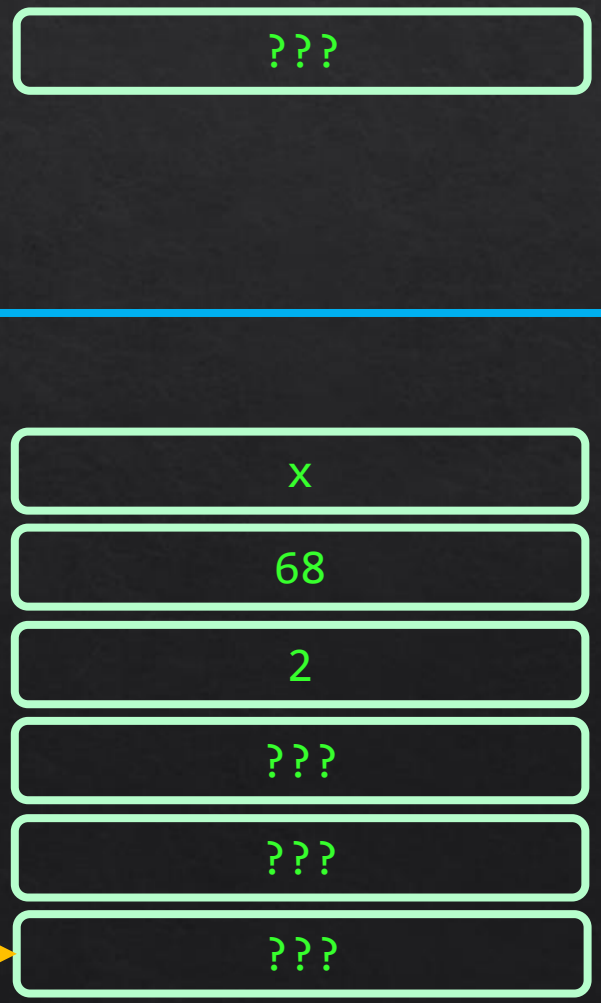
Chall_1.S

- func

Register : Value

SP	
PC	
w0	3
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

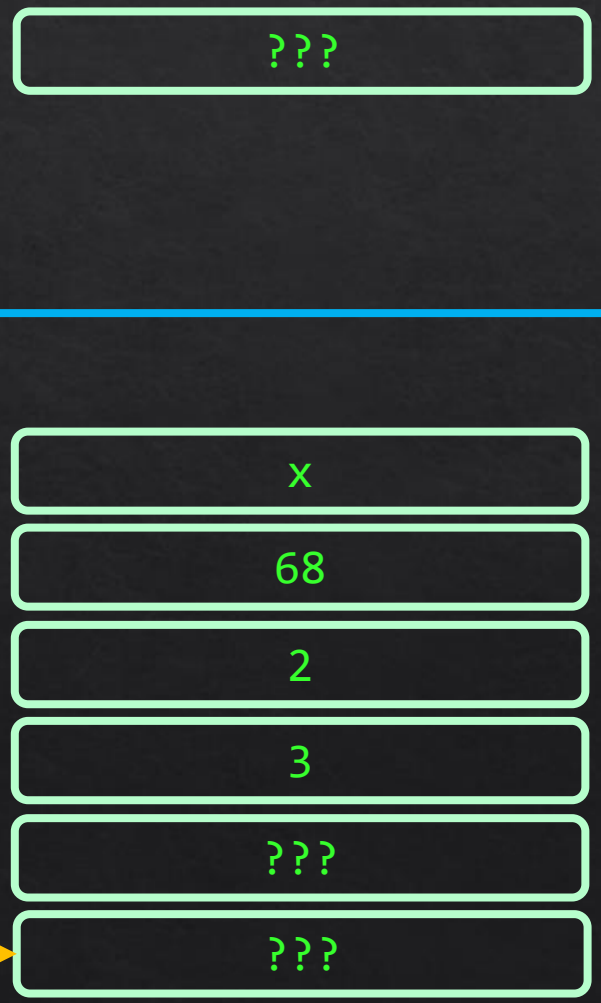
Chall_1.S

- func

Register : Value

SP	
PC	
w0	3
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

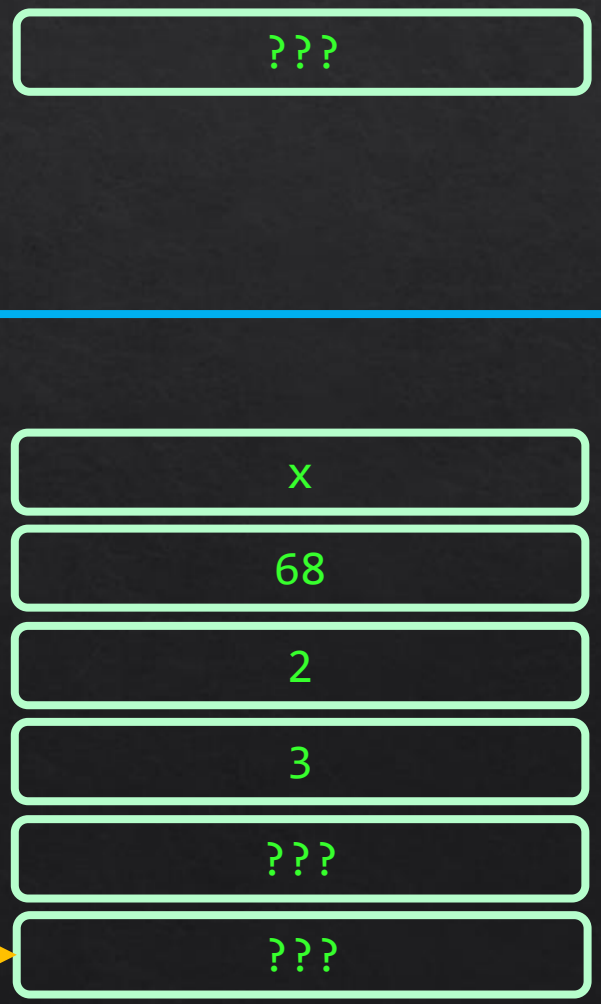
Chall_1.S

- func

Register : Value

SP	
PC	
w0	2
w1	y

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

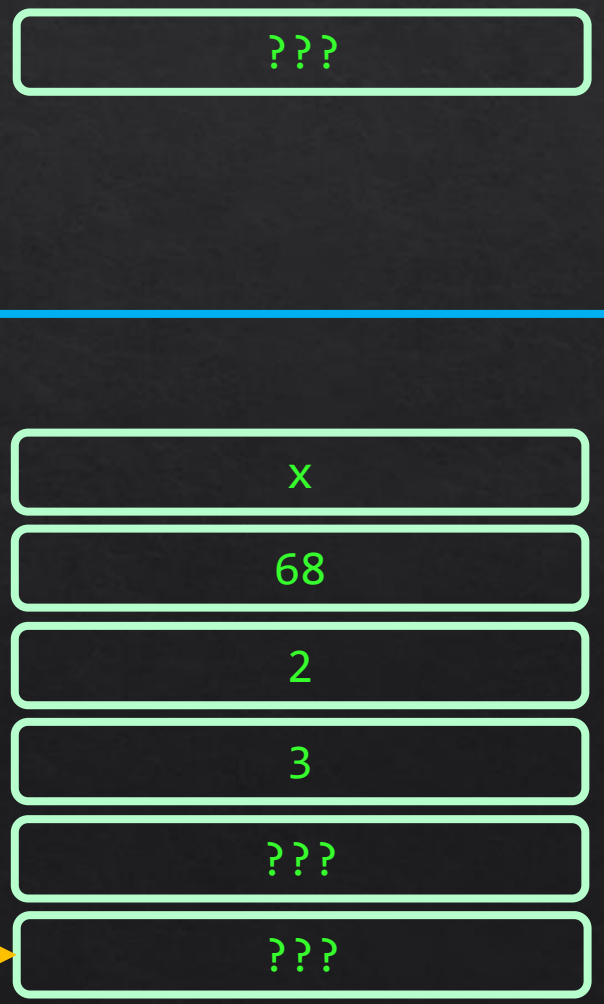
Chall_1.S

- func

Register : Value

SP	
PC	
w0	2
w1	68

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

w0 = 68 << 2 = 272

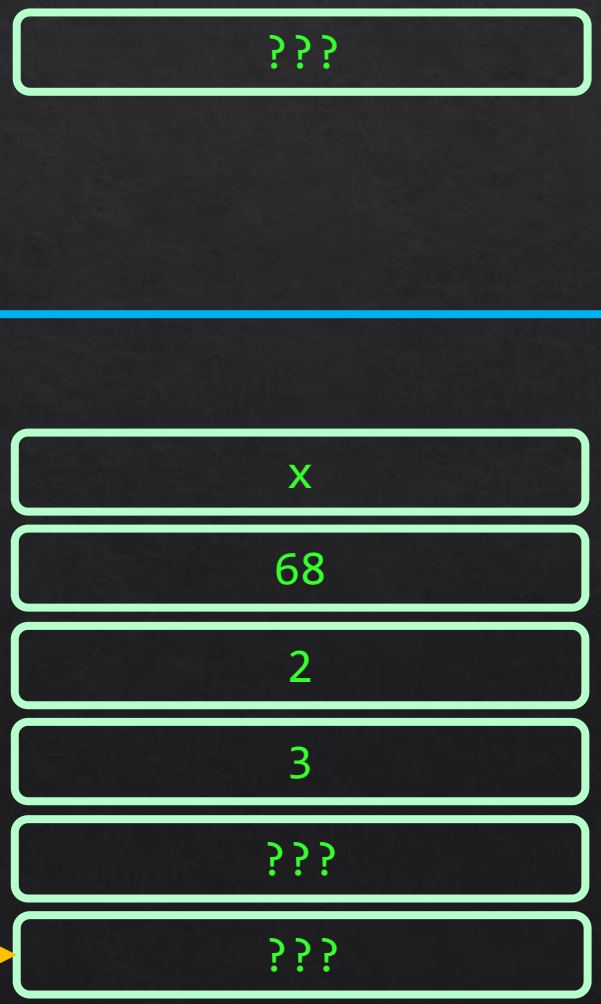
Chall_1.S

- func

Register : Value

SP	
PC	
w0	272
w1	68

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

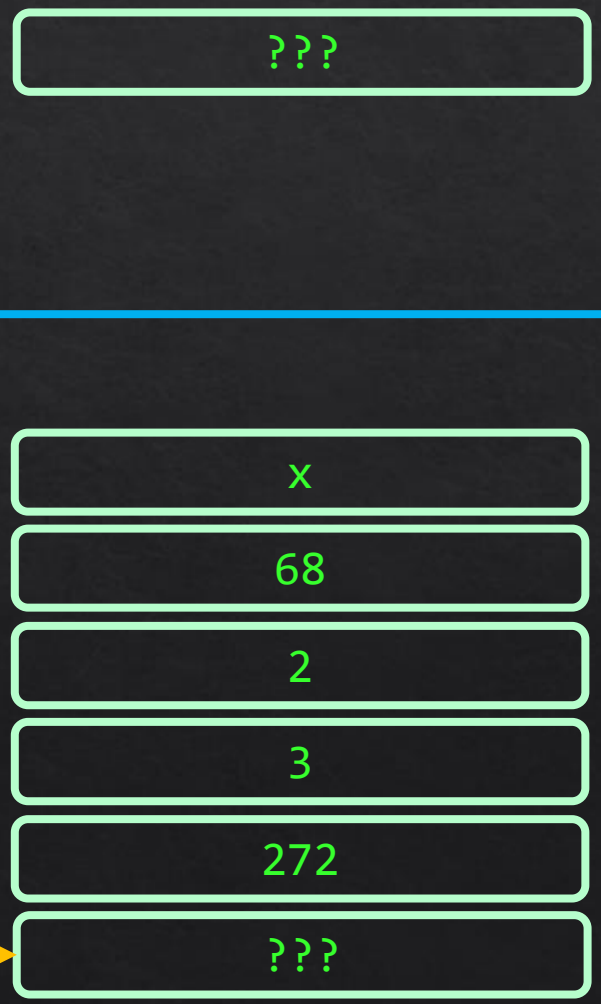
Chall_1.S

- func

Register : Value

SP	
PC	
w0	272
w1	68

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

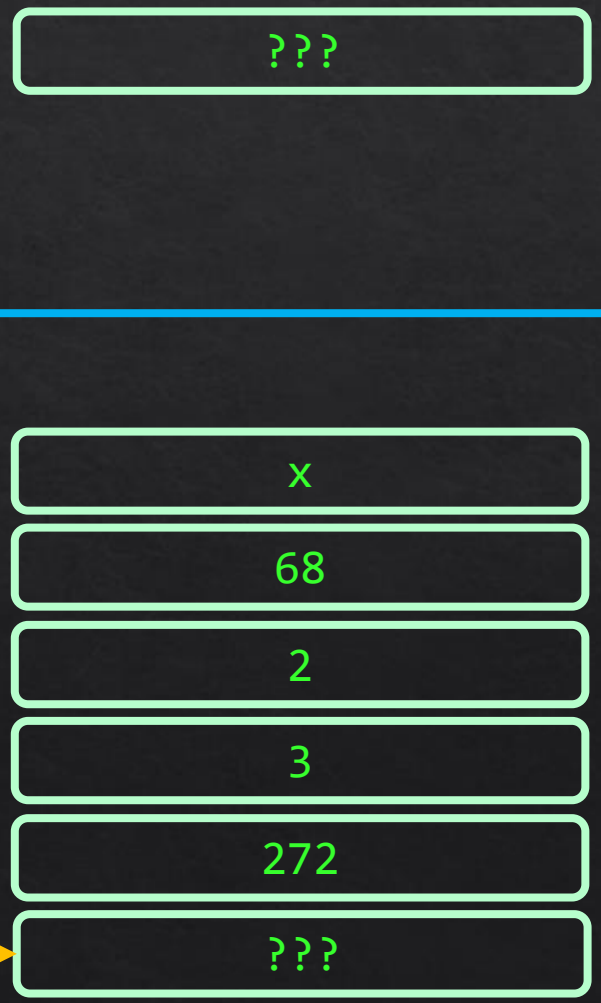
Chall_1.S

- func

Register : Value

SP	
PC	
w0	272
w1	272

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

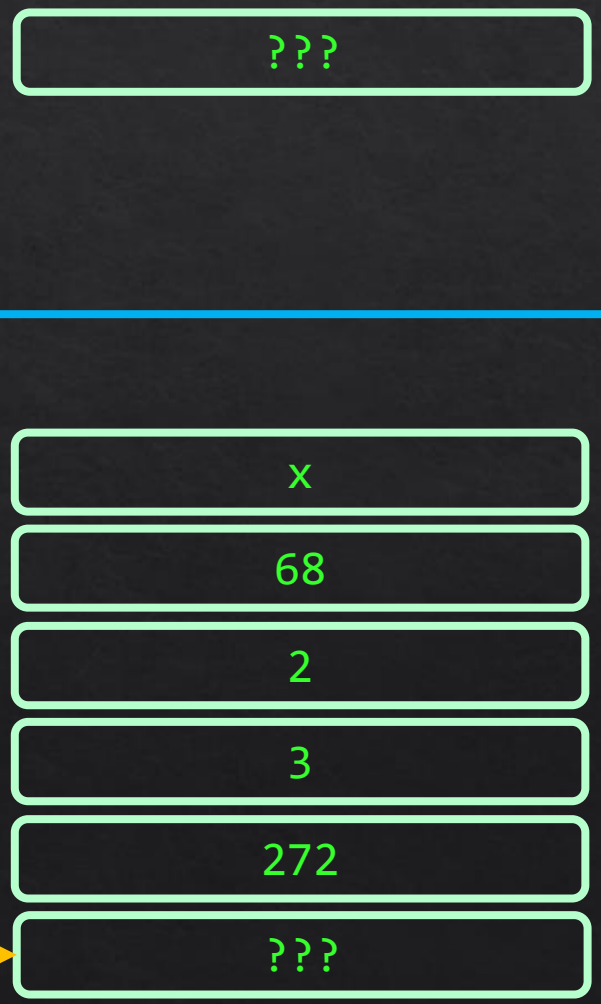
Chall_1.S

- func

Register : Value

SP	
PC	
w0	3
w1	272

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

Chall_1.S

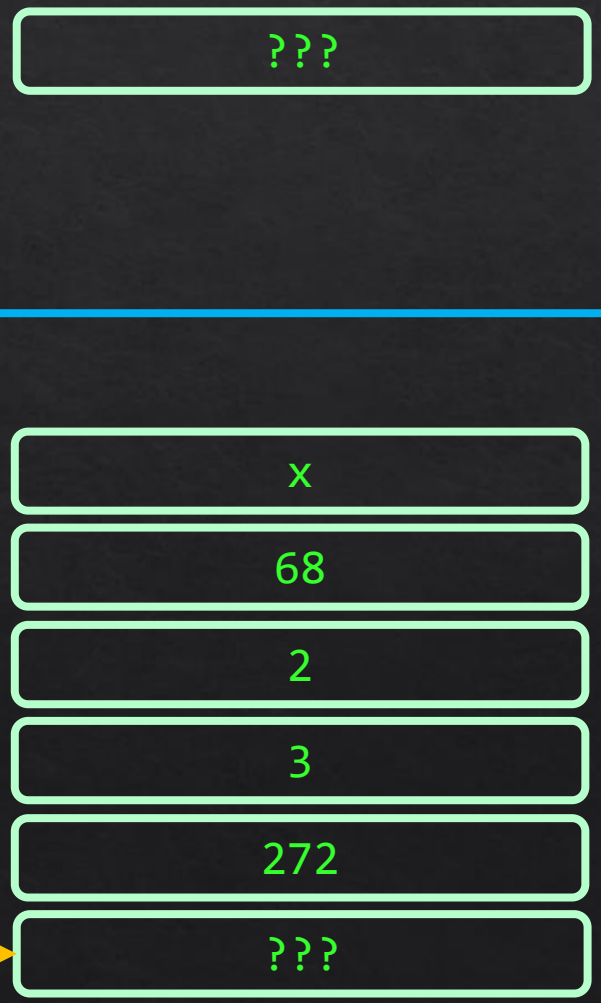
`w0 = 272 / 3 = 90`

- func

Register : Value

SP	
PC	
w0	90
w1	272

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

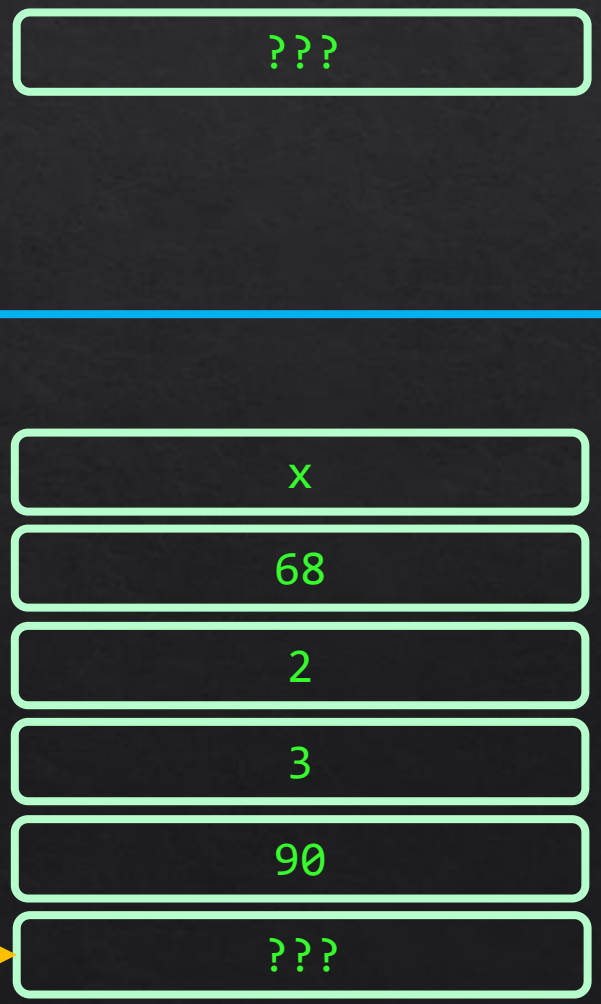
Chall_1.S

- func

Register : Value

SP	
PC	
w0	90
w1	272

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

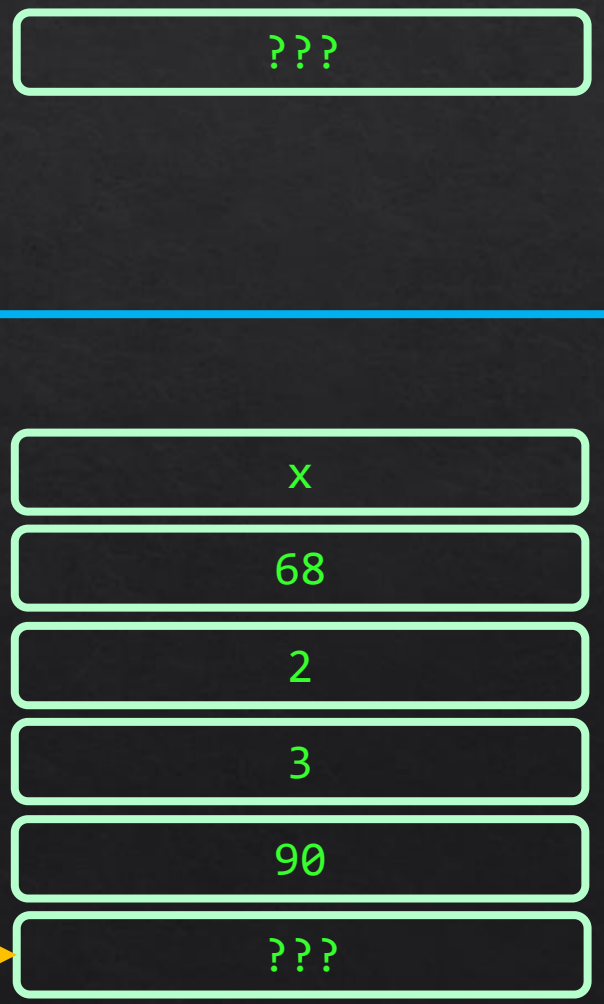
Chall_1.S

- func

Register : Value

SP	
PC	
w0	90
w1	90

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

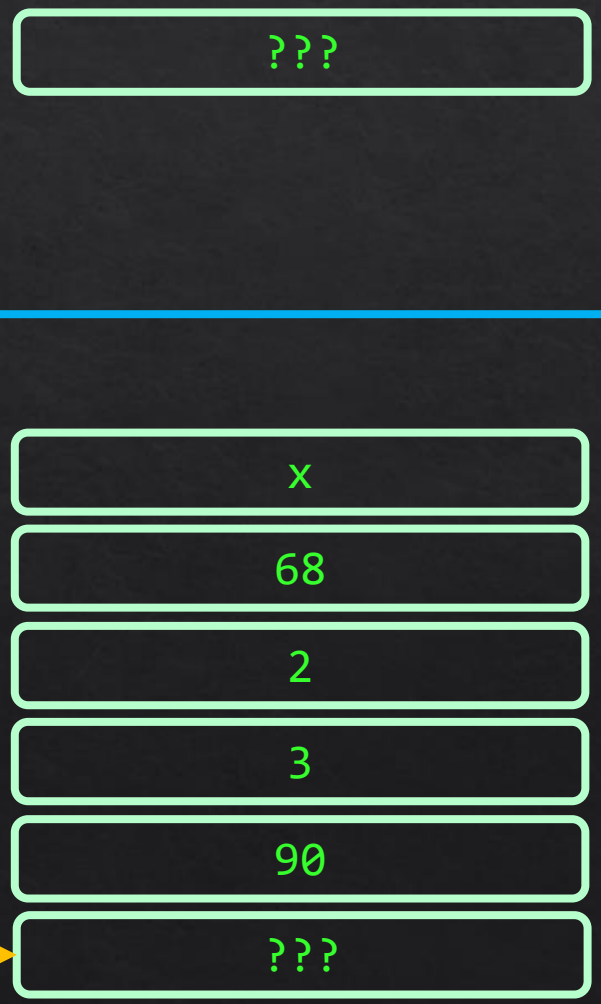
Chall_1.S

- func

Register : Value

SP	
PC	
w0	x
w1	90

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

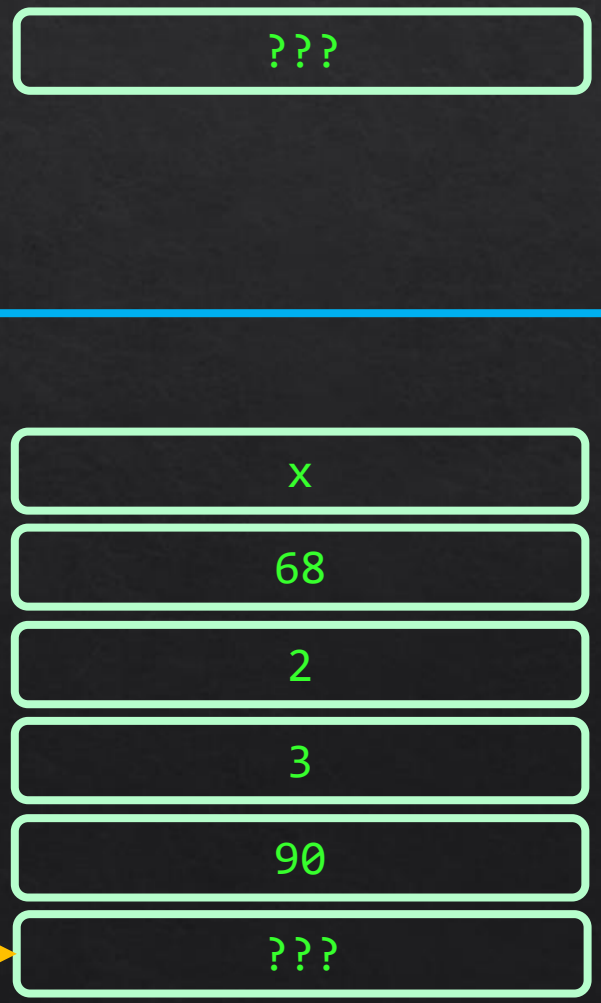
Chall_1.S

- func

Register : Value

SP	
PC	
w0	90-x
w1	90

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

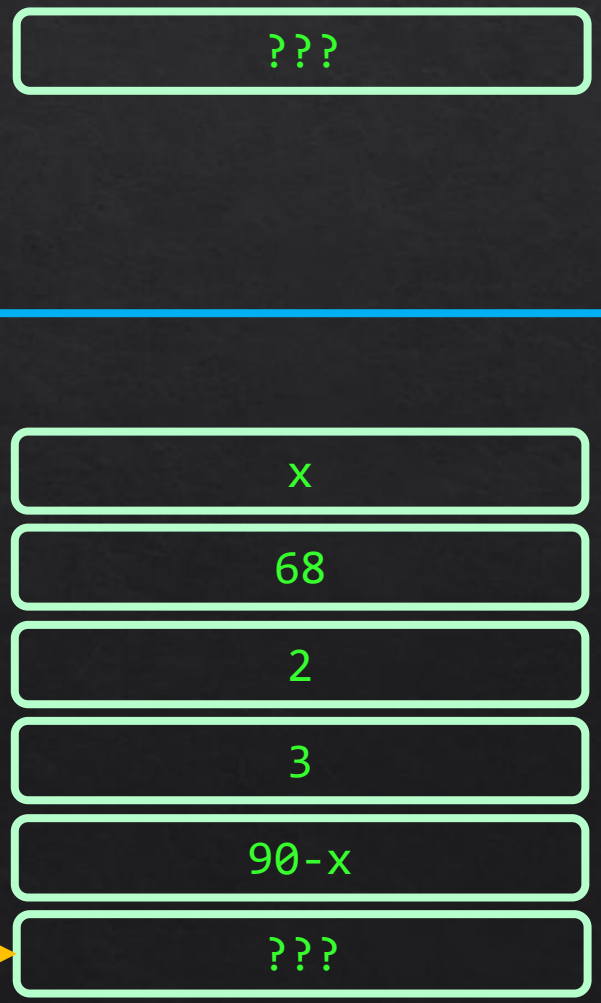
Chall_1.S

- func

Register : Value

SP	
PC	
w0	90-x
w1	90

Stack




```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

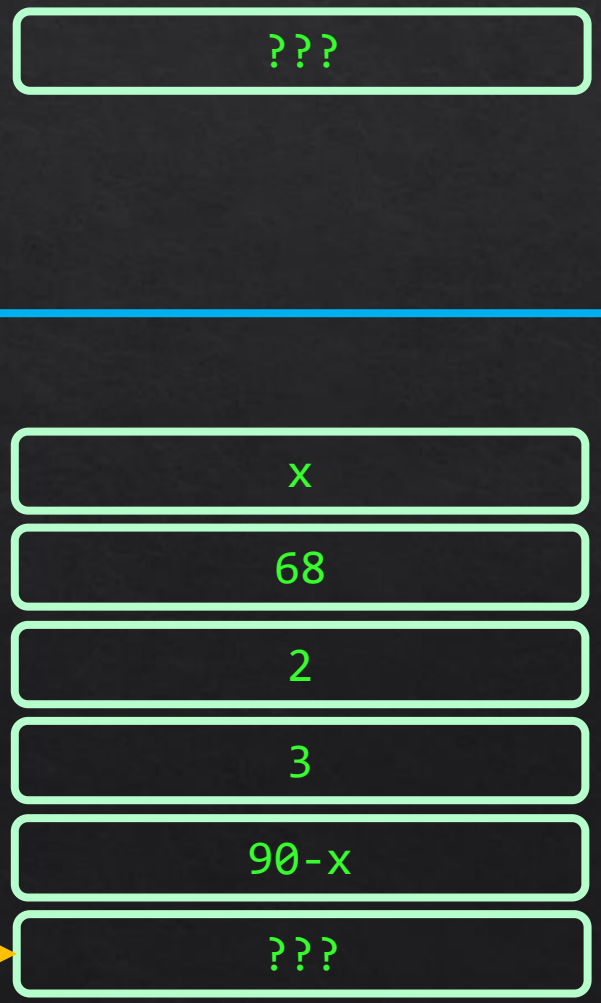
Chall_1.S

- func

Register : Value

SP	
PC	
w0	90-x
w1	90

Stack

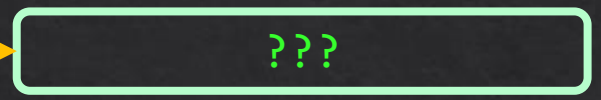



```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

Chall_1.S

- func

Stack



Register : Value

SP	
PC	
w0	90-x
w1	90

```
func:
        sub     sp, sp, #32
        str     w0, [sp, 12]
        mov     w0, 68
        str     w0, [sp, 16]
        mov     w0, 2
        str     w0, [sp, 20]
        mov     w0, 3
        str     w0, [sp, 24]
        ldr     w0, [sp, 20]
        ldr     w1, [sp, 16]
        lsl     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 24]
        sdiv    w0, w1, w0
        str     w0, [sp, 28]
        ldr     w1, [sp, 28]
        ldr     w0, [sp, 12]
        sub     w0, w1, w0
        str     w0, [sp, 28]
        ldr     w0, [sp, 28]
        add     sp, sp, 32
        ret
        .size   func, .-func
        .section        .rodata
        .align  3
```

ret

- `pop  rip`

Chall_1.S

- Main
  - We need x=90 to win.

Register : Value

SP	
PC	
w0	90-x
w1	90

```
main:
        stp     x29, x30, [sp, -48]!
        add     x29, sp, 0
        str     w0, [x29, 28]
        str     x1, [x29, 16]
        ldr     x0, [x29, 16]
        add     x0, x0, 8
        ldr     x0, [x0]
        bl      atoi
        str     w0, [x29, 44]
        ldr     w0, [x29, 44]
        bl      func
        cmp     w0, 0
        bne     .L4
        adrp    x0, .LC0
        add     x0, x0, :lo12:.LC0
        bl      puts
        b       .L6
```

```
cmp 90-x == 0
```

```
x=90=5a
```

Flag

```
picoCTF{0000005a}
```

Tools

GDB, GHIDRA, pwntools

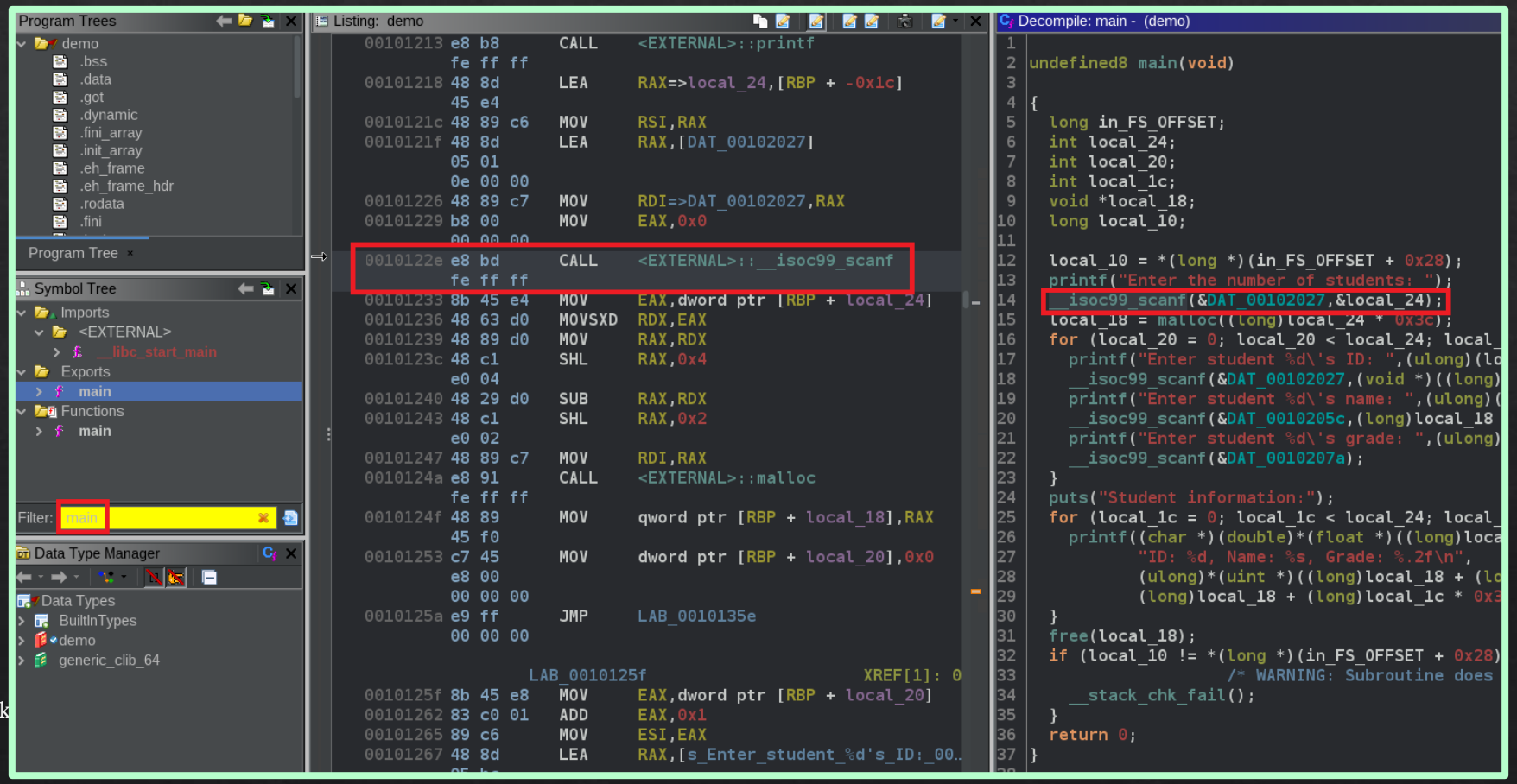
Tools

- 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;
}
```

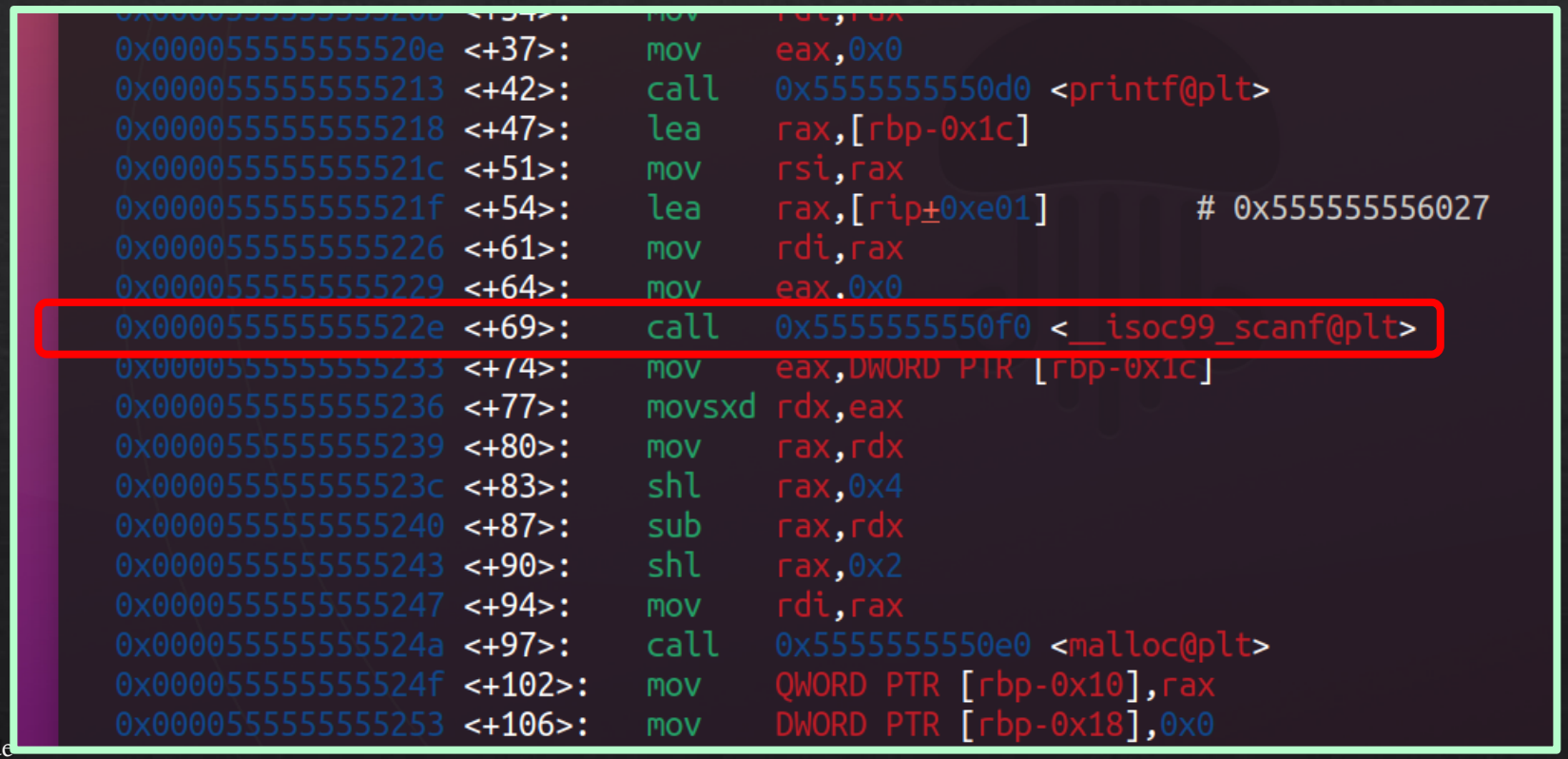
Tools -- GHIDRA

- 22e



Tools -- GDB

- 22e



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 Solution

./reverse_demo_s

Author Sharkkcode

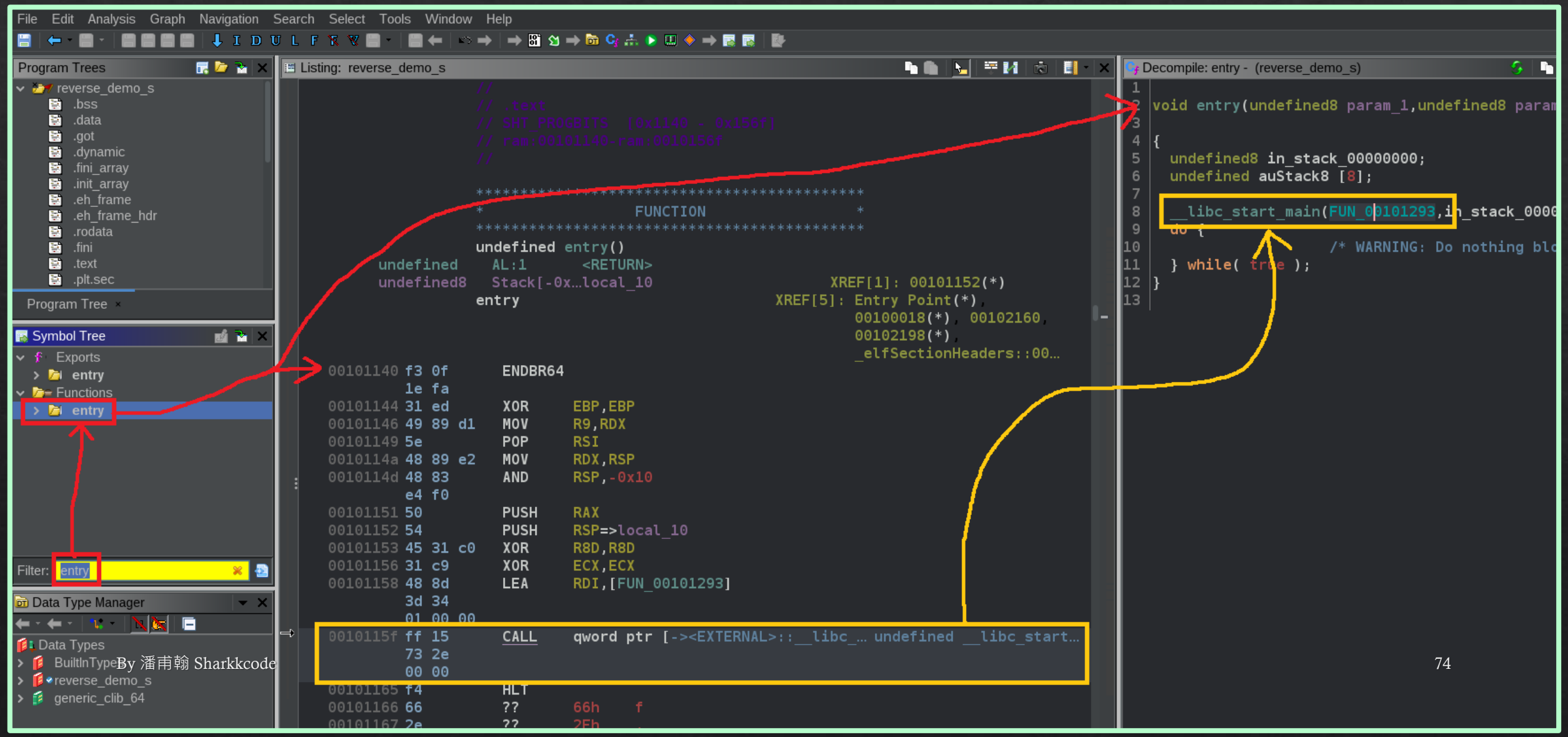
Your goal is to find out the right key!

./reverse_demo_s

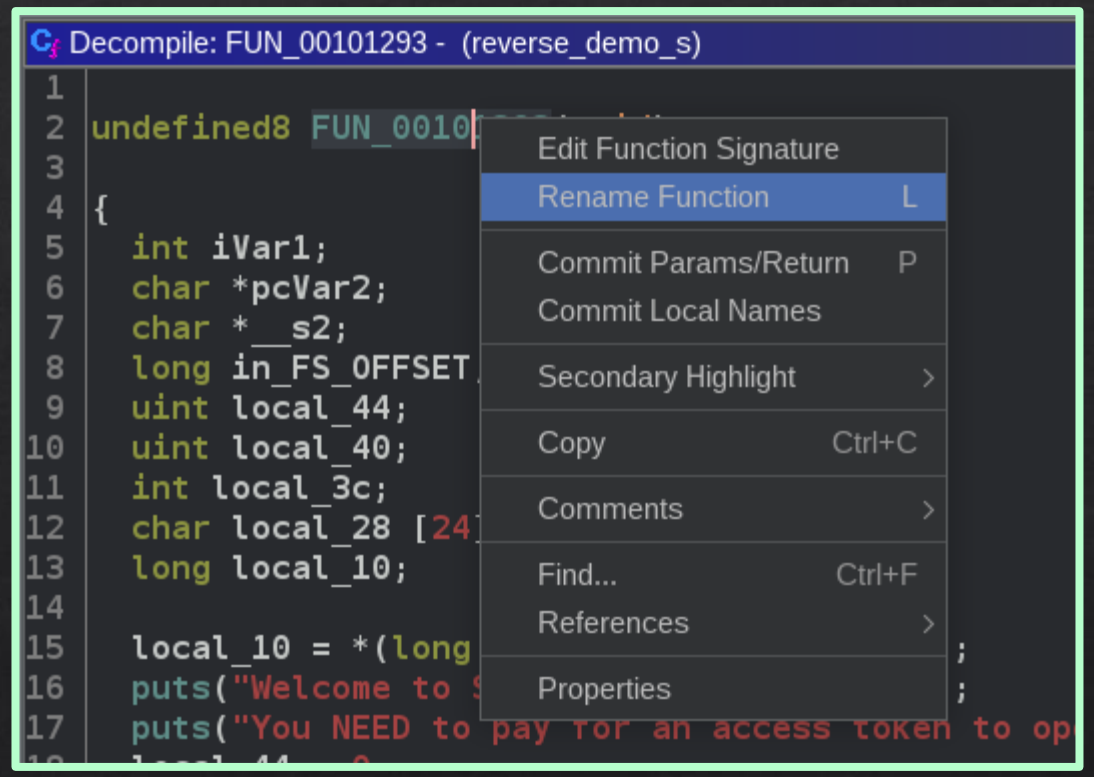
```
~/reverse_ccuisc
~/reverse_ccuisc ./reverse_demo_s
Welcome to Super BIRD Software!!!
You NEED to pay for an access token to open this VALUABLE software~~~
aaa
a
a
a
a
a

aaaa
a
aaaaaa
aaaa

aaa
Enter Access Key: Nahhh! Wrong Key...
Please pay in order to get the right key!!!
Enter Access Key: Nahhh! Wrong Key...
Please pay in order to get the right key!!!
```

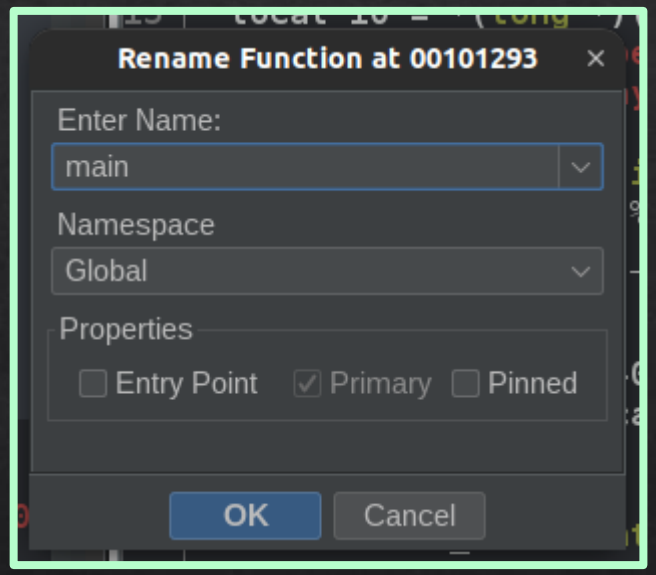
What is going on?

Find main → FUN_00101293

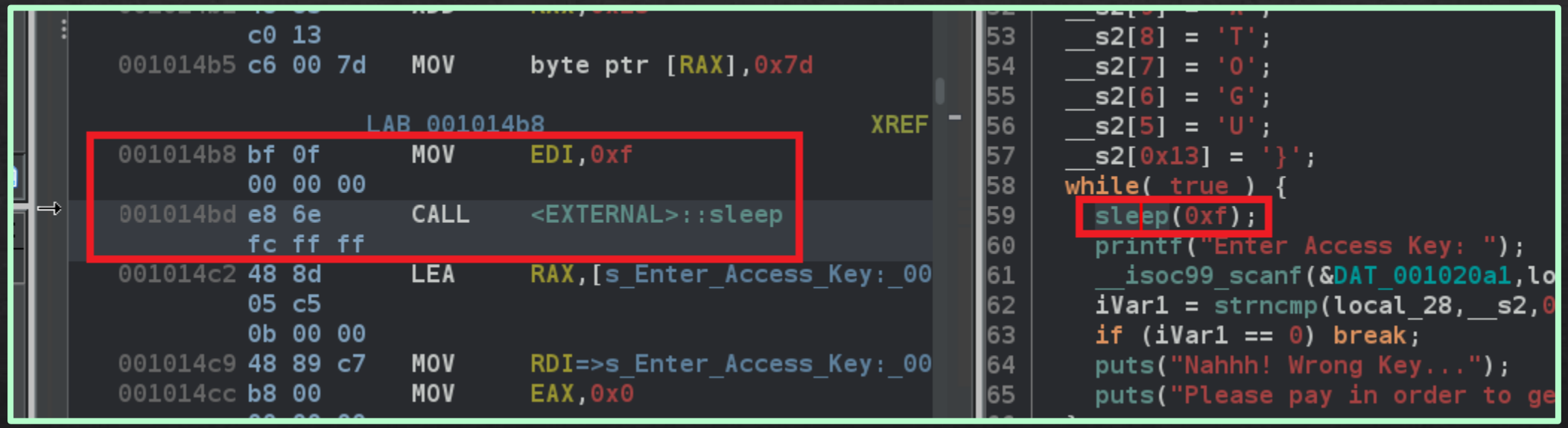


Rename `FUN_00101293` to `main` .

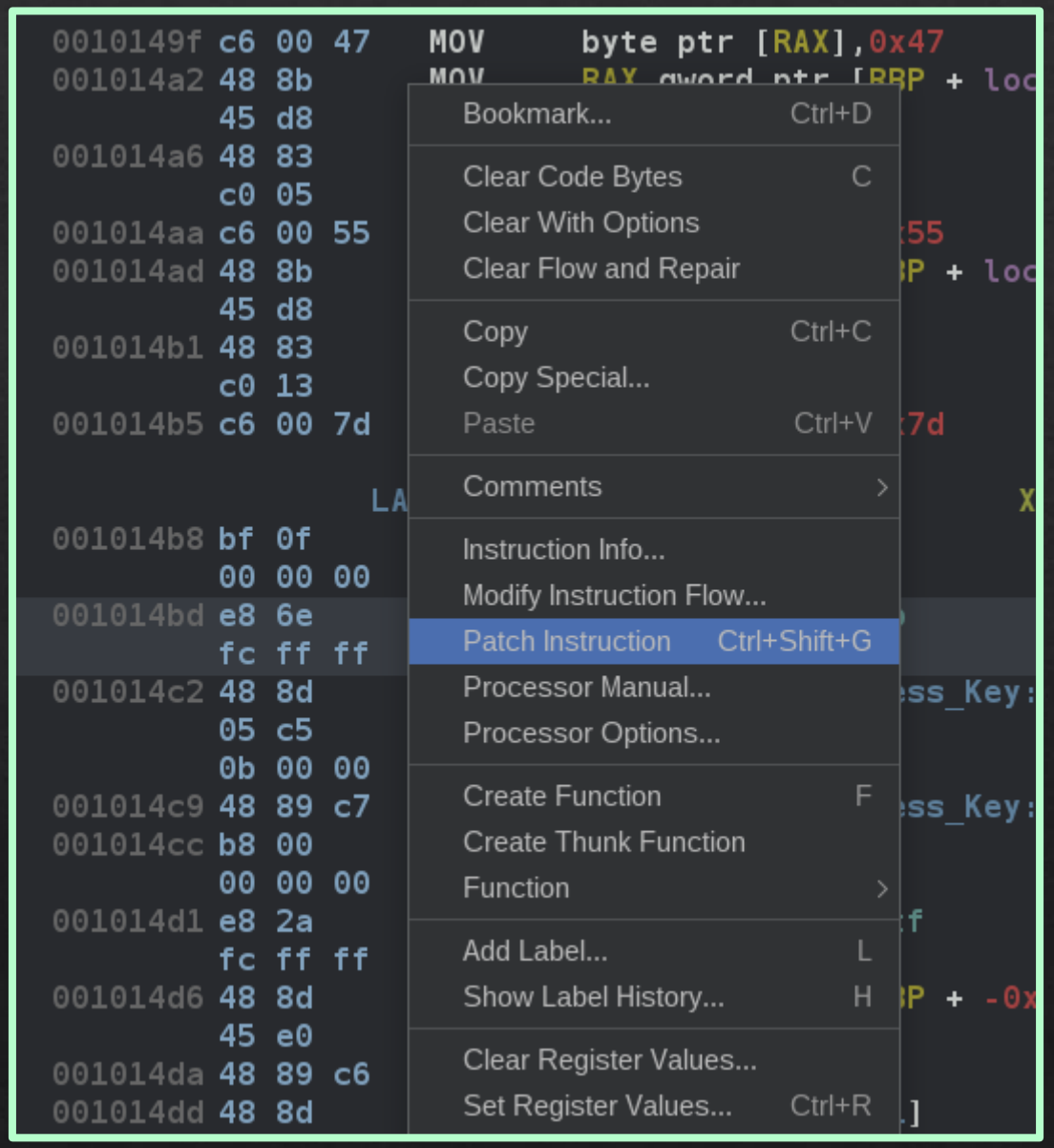




Sleep 15s…


c0 13
001014b5 c6 00 7d MOV byte ptr [RAX],0x7d
LAB_001014b8 XREF
001014b8 bf 0f 00 00 00 MOV EDI,0xf
001014bd e8 6e fc ff ff CALL <EXTERNAL>::sleep
001014c2 48 8d 05 c5 0b 00 00 LEA RAX,[s_Enter_Access_Key:_00
001014c9 48 89 c7 MOV RDI=>s_Enter_Access_Key:_00
001014cc b8 00 MOV EAX,0x0
53 __s2[8] = 'T';
54 __s2[7] = '0';
55 __s2[6] = 'G';
56 __s2[5] = 'U';
57 __s2[0x13] = '}';
58 while(true) {
59 sleep(0xf);
60 printf("Enter Access Key: ");
61 __isoc99_scanf(&DAT_001020a1,lo
62 iVar1 = strncmp(local_28,__s2,0
63 if (iVar1 == 0) break;
64 puts("Nahhh! Wrong Key...");
65 puts("Please pay in order to ge

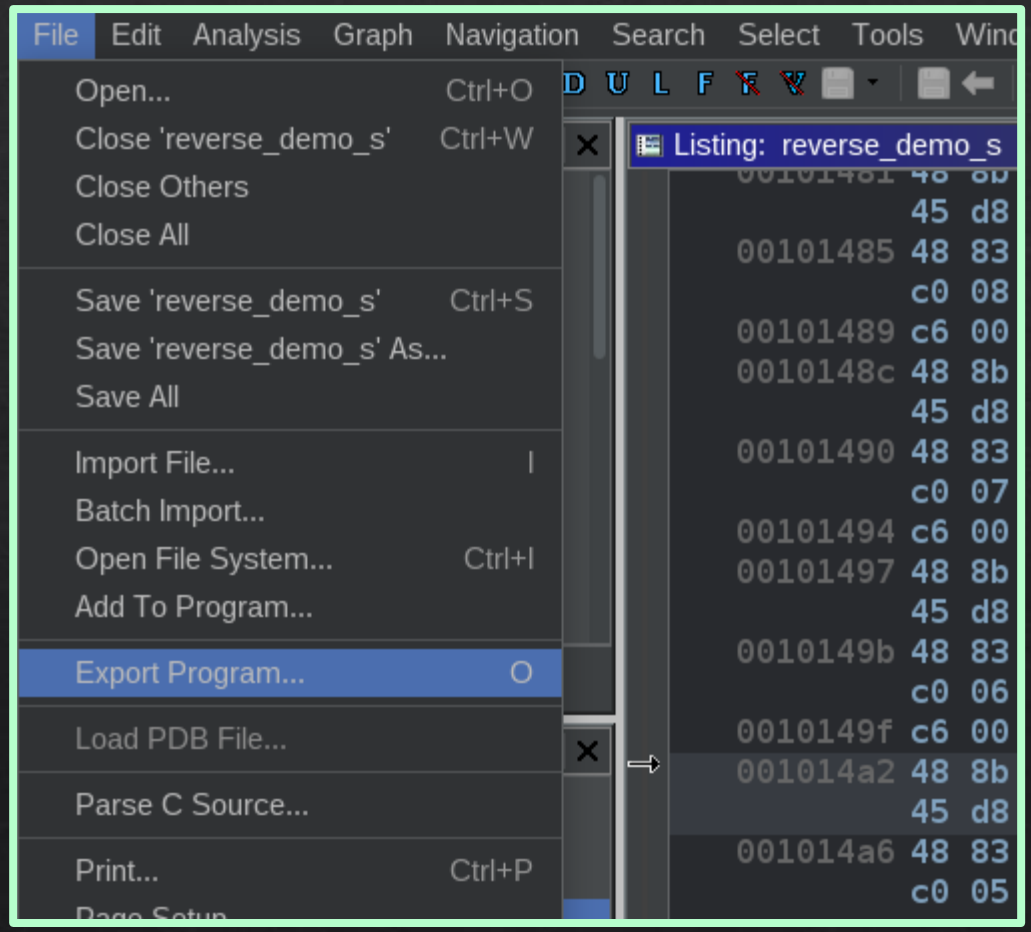
Replace sleep



Replace sleep

```
001014b1 48 83 c0 13    ADD     RAX,0x13
001014b5 c6 00 7d       MOV     byte ptr [RAX],0x7d

                    LAB_001014b8
001014b8 bf 0f 00 00 00 MOV     EDI,0xf
001014bd 90             NOP
001014be 90             NOP
001014bf 90             NOP
001014c0 90             NOP
001014c1 90             NOP
001014c2 48 8d 05 c5    LEA     RAX,[s_Enter_Access_
         0b 00 00
001014c9 48 89 c7       MOV     RDI=>s_Enter_Access_
001014cc b8 00          MOV     EAX,0x0
```

Replace sleep



Replace sleep

```
~/reverse_ccuisc
~/reverse_ccuisc ./reverse_demo_s_nop
Welcome to Super BIRD Software!!!
You NEED to pay for an access token to open this VALUABLE software~~~
Enter Access Key: aaa
Nahhh! Wrong Key...
Please pay in order to get the right key!!!
Enter Access Key:
```

What happens when we enter the right key?

```
58    while( true ) {
59      sleep(0xf);
60      printf("Enter Access Key: ");
61      __isoc99_scanf(&DAT_001020a1,local_28);
62      iVar1 = strncmp(local_28,__s2,0x14);
63      if (iVar1 == 0) break;
64      puts("Nahhh! Wrong Key...");
65      puts("Please pay in order to get the right key!!!");
66    }
67    system("curl parrot.live");
68    printf("That is our secret parrot for that will make your wish come true!!!");
69    if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
70                      /* WARNING: Subroutine does not return */
71      __stack_chk_fail();
72    }
73    return 0;
74  }
75
```

Simple Comparison

```
58    while( true ) {
59      sleep(0xf);
60      printf("Enter Access Key: ");
61      __isoc99_scanf(&DAT_001020a1,local_28);
62      iVar1 = strncmp(local_28,__s2,0x14);
63      if (iVar1 == 0) break;
64      puts("Nahhh! Wrong Key...");
65      puts("Please pay in order to get the right key!!!");
66    }
67    system("curl parrot.live");
68    printf("That is our secret parrot for that will make your wish come true!!!");
69    if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
70                        /* WARNING: Subroutine does not return */
71      __stack_chk_fail();
72    }
73    return 0;
74  }
75
```

`local_28` is user input.

```
__s2[0x12] = *pcVar2;
__s2[0x11] = pcVar2[1];
__s2[0x10] = pcVar2[2];
__s2[0xf] = pcVar2[3];
__s2[0xe] = '!';
__s2[0xd] = 'D';
__s2[0xc] = 'R';
__s2[0xb] = 'I';
__s2[10] = 'B';
__s2[4] = '{';
__s2[3] = 'G';
__s2[2] = 'A';
__s2[1] = 'L';
*__s2 = 'F';
__s2[9] = 'A';
__s2[8] = 'T';
__s2[7] = 'O';
__s2[6] = 'G';
__s2[5] = 'U';
__s2[0x13] = '}';
```

```
__s2 = “FLAG{UGOTABIRD!” + pcVar2[3] + pcVar2[2] + pcVar2[1] +
pcVar2[0] + “}”
```

Line 18 ~ Line 31

local_44 → pcVar2

```
18   local_44 = 0;
19   for (local_40 = 0; (int)local_40 < 0x14; local_40 = local_40 + 1) {
20     if ((int)local_40 % 3 == 0) {
21       local_44 = local_44 + 1;
22     }
23     else {
24       if ((int)local_40 % 3 == 1) {
25         local_44 = local_44 ^ local_40;
26       }
27       else {
28         local_44 = (int)(local_44 * local_40 + -1) / 3;
29       }
30     }
31   }
32   malloc(0x50);
33   pcVar2 = (char *)FUN_00101229(local_44,0x10);
```

Calculate local_44

exp.c

```c
#include <stdio.h>

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

    int local_44 = 0;
    for(int local_40 = 0; local_40 < 0x14; local_40++) {
        if(local_40 % 3 == 0) {
            local_44++;
        } else {
            if(local_40 % 3 == 1) {
                local_44 ^= local_40;
            } else {
                local_44 = (local_44 * local_40 - 1) / 3;
            }
        }
    }

    printf("local_44: %d\n", local_44);

    return 0;
}

```

local_44 → pcVar2

```
~/reverse_ccuisc
~/reverse_ccuisc ./exp
local_44: 4520
~/reverse_ccuisc
```

(char *)FUN_00101129(local_44, 0x10) → pcVar2

Line 33

```
18  local_44 = 0;
19  for (local_40 = 0; (int)local_40 < 0x14; local_40 = local_40 + 1) {
20    if ((int)local_40 % 3 == 0) {
21      local_44 = local_44 + 1;
22    }
23    else {
24      if ((int)local_40 % 3 == 1) {
25        local_44 = local_44 ^ local_40;
26      }
27      else {
28        local_44 = (int)(local_44 * local_40 + -1) / 3;
29      }
30    }
31  }
32  malloc(0x50);
33  pcVar2 = (char *)FUN_00101229(local_44,0x10);
```

(char *)FUN_00101129(local_44, 0x10) → pcVar2

Decompile: FUN_00101229 - (reverse_demo_s)

```
1
2  undefined * FUN_00101229(int param_1,int param_2)
3
4  {
5    int local_1c;
6    int local_c;
7
8    local_1c = param_1;
9    for (local_c = 0x1e; (local_1c != 0 && (local_c != 0)); local_c = local_c + -1) {
10     (&DAT_00104040)[local_c] = "0123456789abcdef"[local_1c % param_2];
11     local_1c = local_1c / param_2;
12   }
13   return &DAT_00104040 + (local_c + 1);
14 }
15
```

Guess???

0x10 = 16
0123456789abcdef → Hexadecimal?

```
>>>
>>> print(hex(4520))
0x11a8
>>>
```

Try…

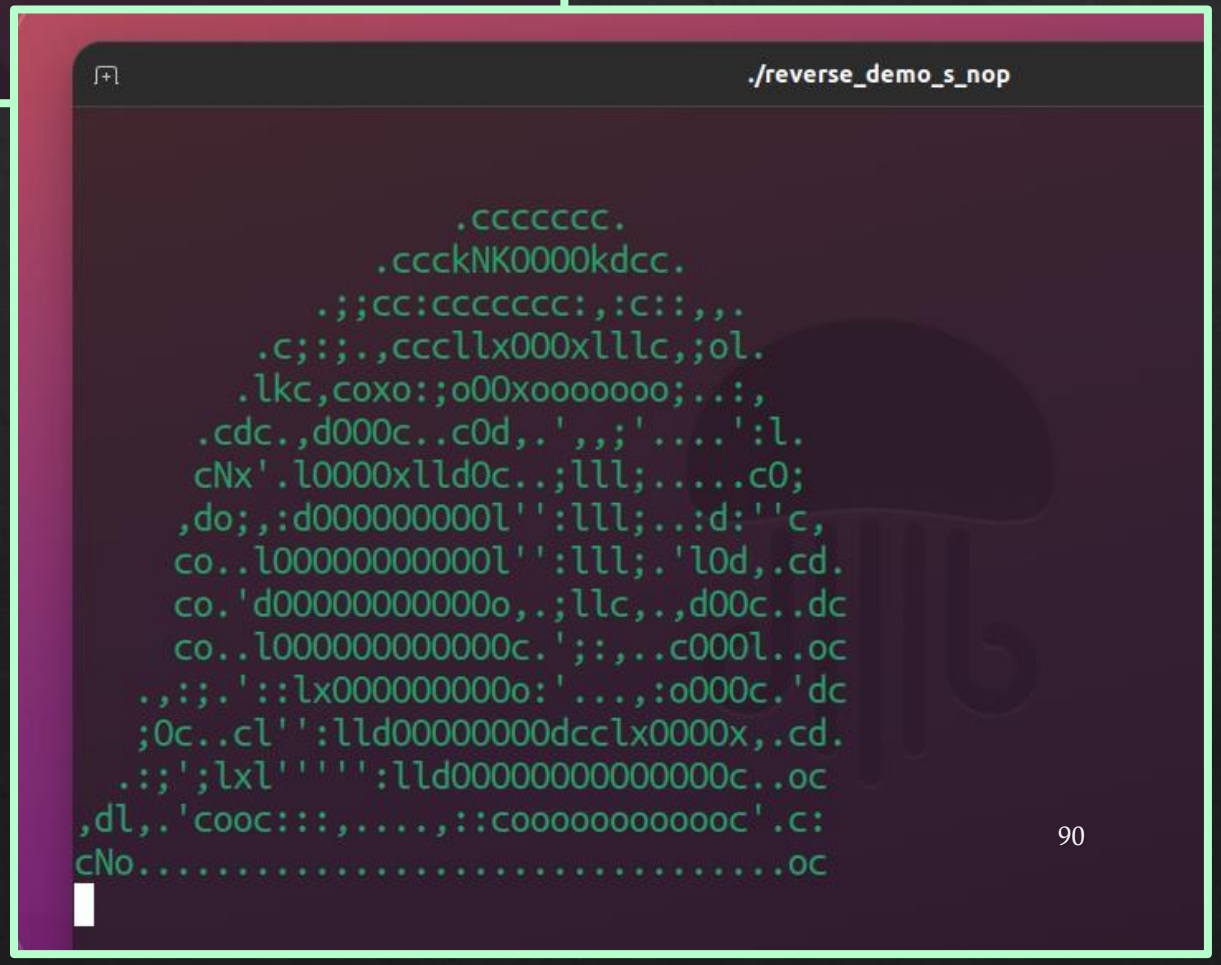
```
pcVar2 = “11a8”
__s2 = “FLAG{UGOTABIRD!” + pcVar2[3] + pcVar2[2] + pcVar2[1] + pcVar2[0] + “}”
```

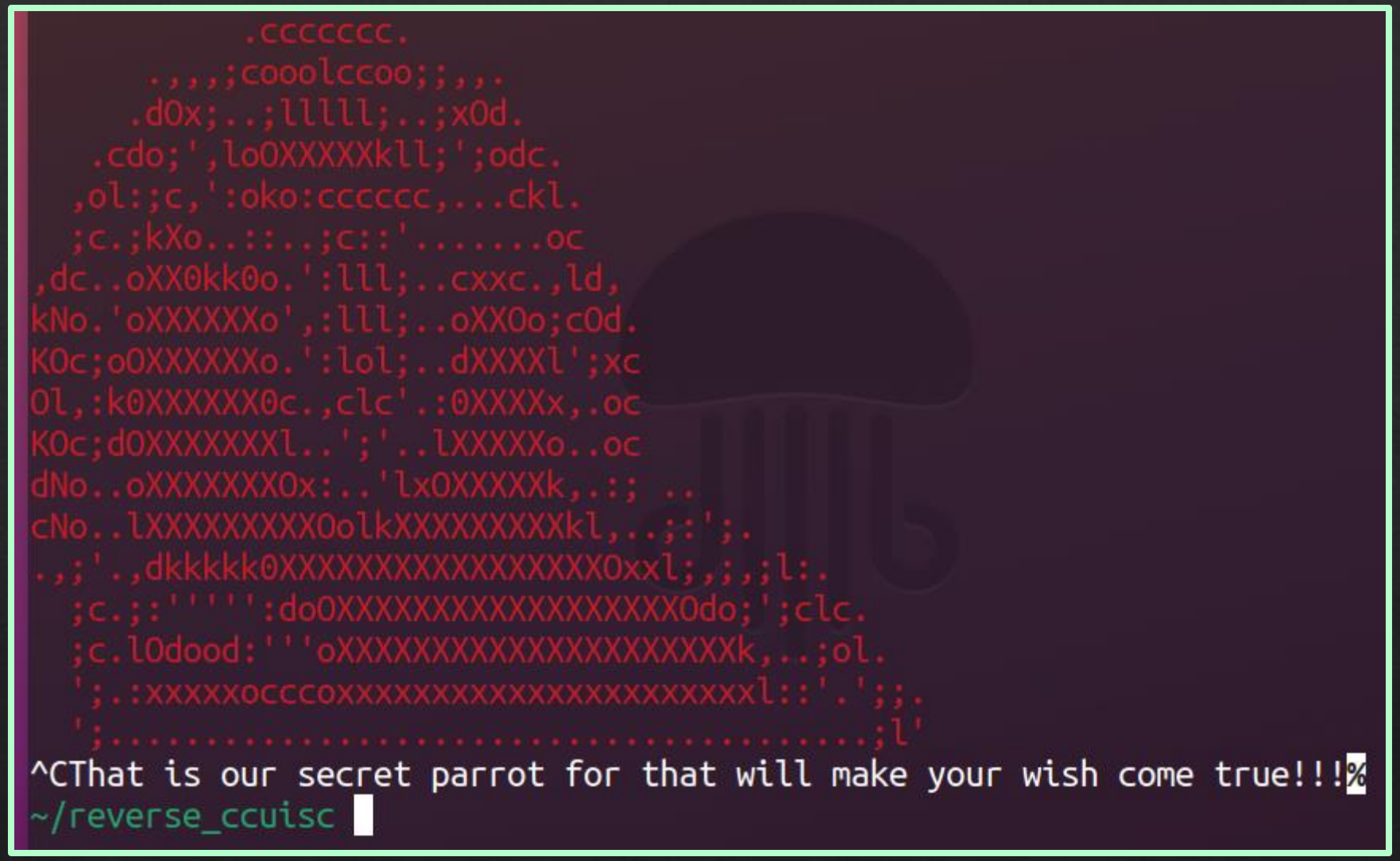


```
__s2 = “FLAG{UGOTABIRD!8a11}”
```

BIRD!!!

```
~/reverse_ccuisc
~/reverse_ccuisc ./reverse_demo_s_nop
Welcome to Super BIRD Software!!!
You NEED to pay for an access token to open this VALUABLE software~~~
Enter Access Key: FLAG{UGOTABIRD!8a11}
```




.ccccccc.
.,,,;cooolccoo;;,,.
.dOx;..;llllll;..;xOd.
.cdo;',loOXXXXXkll;';odc.
,ol:;c,':oko:cccccc,...ckl.
;c.;kXo..::..;c::'.......oc
,dc..oXX0kk0o.':lll;..cxxc.,ld,
kNo.'oXXXXXXo',:lll;..oXXOo;cOd.
KOc;oOXXXXXXo.':lol;..dXXXXl';xc
Ol,:k0XXXXXX0c.,clc'.:0XXXXx,.oc
KOc;dOXXXXXXXl..';'..lXXXXXo..oc
dNo..oXXXXXXXOx:..'lxOXXXXXk,.:; ..
cNo..lXXXXXXXXXOolkXXXXXXXXXkl,..;:';.
.,;'.,dkkkkk0XXXXXXXXXXXXXXXXXOxxl;,;,;l:.
;c.;:'''''':doOXXXXXXXXXXXXXXXXXXOdo;';clc.
;c.lOdood:'''oXXXXXXXXXXXXXXXXXXXXk,..;ol.
';.:xxxxxocccoxxxxxxxxxxxxxxxxxxxxxl::'.';;.
';.....;l'
^CThat is our secret parrot for that will make your wish come true!!!%
~/reverse_ccuisc

Lab

./tea

From SHELL CTF 2022

Your goal is to find out the FLAG!

Lab Solution

./tea

From SHELL CTF 2022

Your goal is to find out the FLAG!

tea

- main
  - len == 20
  - Some functions.

```
1   undefined8 main(void)
2   {
3   size_t sVar1;
4
5   boilWater();
6   sVar1 = strlen(pwd);
7   if (sVar1 == 0x20) {
8       addSugar();
9       addTea();
10      addMilk();
11      strainAndServe();
12  }
13  else {
14      puts("wrong length");
15  }
16  return 0;
17  }
```


My SHELL CTF 2022 writeup: https://sharkkcode.github.io/2022/08/16/SHELL-CTF-2022/

tea

- boilWater
  - Enter flag.

```
1  void boilWater(void)
2  {
3  printf("Enter the flag: ");
4  __isoc99_scanf(&DAT_00102019,pwd);
5  return;
6  }
```


My SHELL CTF 2022 writeup: https://sharkkcode.github.io/2022/08/16/SHELL-CTF-2022/

tea

- strainAndServe
  - Compare flag.

```
1   void strainAndServe(void)
2   {
3   int iVar1;
4
5   iVar1 = strcmp("R;crc75ihl`cNYe`]m%50gYhugow~34i",pwd);
6   if (iVar1 == 0) {
7       puts("yep, that\'s right");
8   }
9   else {
10      puts("nope, that\'s not it");
11  }
12  return;
13  }
```


My SHELL CTF 2022 writeup: https://sharkkcode.github.io/2022/08/16/SHELL-CTF-2022/

tea

- How the flag is modified?

```
flag ---- [addSugar] ---> flag2
flag2 --- [addTea] -----> flag3
flag3 --- [addMilk] ----> flag4

flag4 is R;crc75ihl`cNYe`]m%50gYhugow~34i
```


My SHELL CTF 2022 writeup: https://sharkkcode.github.io/2022/08/16/SHELL-CTF-2022/

tea

- How the flag is modified? (more detailed)

```
part1: String with no '5'.
part2: String starts with '5' and with no 'R'
part3: String starts with 'R'

flag4 = ( part3 + part1 + part2 )
```


My SHELL CTF 2022 writeup: https://sharkkcode.github.io/2022/08/16/SHELL-CTF-2022/

tea

- How to generate the initial flag?

> Notice that **flag4** is just simply move part3 to the front. The length of part2 is knowned. If we know the length of part3, we can calculate the length of part1 immediately. **So we can brute force part3's length.**

> Why the length of part2 is knowned? Because (part1 + part2) is not modified. If we found a '5' in (part1 + part2), that '5' must be the start of part2. A the end of part2 is the end of **flag4** , obviously.

My SHELL CTF 2022 writeup: https://sharkkcode.github.io/2022/08/16/SHELL-CTF-2022/

THANKS

Q&A