lab01.c

#include <stdio.h>

int helloWorld() {

    int j = 0;

    while (j < 4) {

        printf("HELLO\n");

        j++;

    }

    return 0;

}

int main(int argc, char\*\* argv) {

    helloWorld();

}

lab01\_out.txt (line by line explanation highlighted in yellow

lab01.o: file format elf64-x86-64

Disassembly of section .text:

0000000000000000 <helloWorld>: helloWorld function here

0: 55 push %rbp push rbp value onto stack

1: 48 89 e5 mov %rsp,%rbp move rbp value to rsp register

4: 48 83 ec 10 sub $0x10,%rsp subtract $0x10 from rsp

8: c7 45 fc 00 00 00 00 movl $0x0,-0x4(%rbp) conditional move of rbp

f: eb 10 jmp 21 <helloWorld+0x21> jump to different address

11: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 18 <helloWorld+0x18> address replace

18: e8 00 00 00 00 callq 1d <helloWorld+0x1d> call helloWorld

1d: 83 45 fc 01 addl $0x1,-0x4(%rbp) add rbp value to $0x1

21: 83 7d fc 03 cmpl $0x3,-0x4(%rbp) compare rbp to $0x3

25: 7e ea jle 11 <helloWorld+0x11> conditional jump to helloWorld

27: b8 00 00 00 00 mov $0x0,%eax move eax value to $0x0

2c: c9 leaveq restore ebp from stack

2d: c3 retq return

000000000000002e <main>:

2e: 55 push %rbp push rbp value onto stack

2f: 48 89 e5 mov %rsp,%rbp move rbp value to rsp register

32: 48 83 ec 10 sub $0x10,%rsp subtract rsp from $0x10

36: 89 7d fc mov %edi,-0x4(%rbp) move rbp value to edi

39: 48 89 75 f0 mov %rsi,-0x10(%rbp) move rbp value to rsi

3d: b8 00 00 00 00 mov $0x0,%eax move eax to $0x0

42: e8 00 00 00 00 callq 47 <main+0x19> call main

47: b8 00 00 00 00 mov $0x0,%eax move eax to $0x0

4c: c9 leaveq restore ebp from stack

4d: c3 retq return

lab01.s (line by line explanation highlighted in yellow)

    .file   "lab01.c" original source file name

    .text declaring the start of code section

    .section    .rodata read only data in this section

.LC0: memory address for this data

    .string "HELLO" the string is “HELLO”

    .text text section

    .globl  helloWorld globally visible function helloWorld

    .type   helloWorld, @function helloWorld is a function

helloWorld: function is named helloWorld

.LFB0: label

    .cfi\_startproc initializing internal data structure

    pushq   %rbp push rbp value onto stack

    .cfi\_def\_cfa\_offset 16 define change of stack pointer offset

    .cfi\_offset 6, -16 stack pointer offset

    movq    %rsp, %rbp copy rbp value to rsp

    .cfi\_def\_cfa\_register 6 stack pointer register

    subq    $16, %rsp subtract 16 from rsp

    movl    $0, -4(%rbp) move -4(%rbp) to register 0

    jmp .L2 jump to L2

.L3: label

    leaq    .LC0(%rip), %rdi load effective address rdi into LC0

    call    puts@PLT program linkage table call

    addl    $1, -4(%rbp) add -4(%rbp) to $1

.L2: label

    cmpl    $3, -4(%rbp) comparing contents of two registers

    jle .L3 conditional jump to L3

    movl    $0, %eax move eax to 0

    leave releasing used stack pointer space

    .cfi\_def\_cfa 7, 8 defining rule for cfa computation

    ret popping return address off stack

    .cfi\_endproc closing previously opened startproc

.LFE0: label

    .size   helloWorld, .-helloWorld setting size for helloWorld

    .globl  main globally visible function main

    .type   main, @function main is a function

main: function is named main

.LFB1: label

    .cfi\_startproc initializing internal data structure

    pushq   %rbp push rbp value onto stack

    .cfi\_def\_cfa\_offset 16 define change of stack pointer offset

    .cfi\_offset 6, -16 stack pointer offset

    movq    %rsp, %rbp copy rbp value to rsp

    .cfi\_def\_cfa\_register 6 stack pointer register

    subq    $16, %rsp subtract $16 content from rsp

    movl    %edi, -4(%rbp) move -4(%rbp) to edi

    movq    %rsi, -16(%rbp) move -16(%rbp) to rsi

    movl    $0, %eax move %eax to $0

    call    helloWorld call helloWorld function

    movl    $0, %eax move %eax to $0

    leave releasing used stack pointer space

    .cfi\_def\_cfa 7, 8 defining rule for cfa computation

    ret popping return address off stack

    .cfi\_endproc closing previously opened startproc

.LFE1: label

    .size   main, .-main setting size of main

    .ident  "GCC: (Ubuntu 7.4.0-1ubuntu1~18.04.1) 7.4.0" gcc leaving trace

    .section    .note.GNU-stack,"",@progbits accommodates non-exec stack